(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
2 August 2001 (02.08.2001)

PCT

(10) International Publication Number
WO 01/55886 A2

(51) International Patent Classification[7]: G06F 17/00

(21) International Application Number: PCT/US01/01944

(22) International Filing Date: 19 January 2001 (19.01.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/177,240	20 January 2000 (20.01.2000)	US
09/765,697	18 January 2001 (18.01.2001)	US
09/766,300	18 January 2001 (18.01.2001)	US
09/766,301	18 January 2001 (18.01.2001)	US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier applications:

US	60/177,240 (CON)
Filed on	20 January 2000 (20.01.2000)
US	09/765,697 (CON)
Filed on	18 January 2001 (18.01.2001)
US	09/766,300 (CON)
Filed on	18 January 2001 (18.01.2001)
US	09/766,301 (CON)
Filed on	18 January 2001 (18.01.2001)

(71) Applicant *(for all designated States except US)*: **PRICERADAR, INC.** [US/US]; Suite 370, 2105 Hamilton Avenue, San Jose, CA 95125 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **WARFIELD, Robert, W.** [US/US]; 3470 Merrill Road, Aptos, CA 95003 (US). **CAIN, Ronald, A.** [US/US]; 1669 Nelson Road #6, Scotts Valley, CA 95066 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/55886 A2

(54) Title: SYSTEM, METHOD AND DATABASE FOR CLASSIFYING PRODUCT INFORMATION OBTAINED FROM A COMPUTER NETWORK

(57) Abstract: A product search system is provided which comprises a logic defining a branched decision tree; logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; logic for classifying product entries having product description with a branch terminus of the decision tree by assigning the product entry to a first tree level of the decision tree, and testing the product descriptions against decision node queries leading from the first tree level until a decision node query leading to a branch terminus is satisfied; and logic for searching the classified product entries.

SYSTEM, METHOD AND DATABASE FOR CLASSIFYING PRODUCT INFORMATION OBTAINED FROM A COMPUTER NETWORK

Background of the Invention

Field of the Invention

The present invention relates to a network-based service that provides a benefit to the consumer by searching, locating, and aggregating products on the Internet. More particularly, the present invention relates to an Internet-based service that provides a consumer with an aggregated resource of identical products available on the Internet, as well as, valuation information associated with each available identical product.

Background

There has recently been a tremendous growth in the number of computers connected to the Internet. A client computer connected to the Internet can download various types of information from server computers using client application software, which typically accepts commands from a user and obtains data and services by sending requests to server applications running on the server computers. A number of protocols may be used to exchange commands and data between computers connected to the Internet. For instance, the protocols may include the File Transfer Protocol (FTP), the Hyper Text Transfer Protocol (HTTP), the Simple Mail Transfer Protocol (SMTP), and the Gopher document protocol. Typically, the HTTP protocol is used to access data on the World Wide Web, often referred to as the Internet or "the Web". The Internet is an information service that provides electronic documents and information, as well as links between electronic documents and information. It is made up of numerous Web sites located around the world that maintain and distribute electronic documents. A Web site may use one or more server computers that store and distribute documents in a number of formats, including the Hyper Text Markup Language (HTML). An HTML document contains text and metadata (commands providing formatting information), as

well as embedded links that reference other data or documents, graphics, video, or any combination thereof.

A Web browser is a client application or, preferably, an integrated operating system utility that communicates with server computers via FTP, HTTP, or other communication protocols. Web browsers retrieve electronic documents from the network and present them to a user. Web browsers receive content from a server sent over the Internet that is typically encoded in Hyper Text Markup Language (HTML) and executed by the browser on a client computer. Generally, web browsers also typically support the usage of additional components such as Java Applets, ActiveX Controls, and Plug-Ins to provide extra functionality. These additional components, commonly referred to as "client bits," are typically stored as executables in the memory of the client computer, and can be installed onto the client computer directly from a storage medium or downloaded from a server over the Internet. The functional components such as Java Applets, ActiveX Controls, and Plug-Ins are mapped into the script so that actions, methods, or properties of an object can be called therefrom. (ActiveX Controls are reusable software components that incorporate ActiveX technology, which enables software applications to interact with one another in a networked environment regardless of the language in which the components were created. ActiveX Controls can be embedded in Web pages to produce animation and other multimedia effects, interactive objects and sophisticated applications. ActiveX Controls can be written in a variety of programming languages, including C, C++, Visual Basic, and Java. A Plug-In, on the other hand, is a software component designed to plug into the Netscape Navigator browser, and to permit the browser to access and execute files embedded in HTML documents that are in formats the browser normally would not normally recognize.)

Moreover, Web browsers typically contain an associated scripting space, which is memory space allocated for a browser instance, for the reception of electronic data called a script. Web browsers receive scripts from the network into the scripting space and execute instructions contained in the script. One such instruction contained in a script might be presenting data to a user, usually

by way of an output device such as a computer monitor. In addition to data for presentation to a user, the script may also contain mappings to objects and services stored in the memory of a client computer and instructions for interaction with or communication to and from those objects and services. A script might also contain additional instructions as well. An exchange between script instructions in a scripting space and a service or object can be facilitated by additional objects, such as a Plug-In or ActiveX control. In these cases, a mapping to the Plug-In or ActiveX control is contained in the script, and the Plug-In or ActiveX control performs some operation towards carrying out the script instruction.

Within the Internet environment, the term "search engine" is often used to generically describe both true search engines and directories, although they are generally not the same. Search engines typically create their listings automatically by "searching or crawling" the Web. A directory, on the other hand, depends on humans for its listings, i.e., a person submits a short description for an entire site or editors write a description for sites they review.

Search engines typically include a "crawler" (also referred to as a "spider" or "bot") that visits a Web page or URL, reads it, and then follows links to other pages within the site. The crawler returns to the site on a regular basis to look for changes. Everything the crawler finds generally goes into an index, which is another part of the search engine. The index may be viewed as a file or container holding a copy of every Web page that the crawler finds. The primary purpose of the index is to provide a way to quickly look up a document URL based on words specified in a query. If a Web page changes, then the index is updated with new information. The search engine software, which is yet another part of the search engine, is a program that sifts through the pages recorded in the index to find documents fulfilling a search query submitted by a user.

Once the search engine is given a set of start addresses and restriction rules
(e.g., instructions), a crawler can retrieve documents following all recursive links from the documents that correspond to the start addresses that pass the

restriction rules. For example, a crawler may recursively follow all links from the documents that correspond to specified start addresses, limiting the URL space by filtering out those that do not pass the specified crawl restriction rules. The primary application of the crawler is to build an index of a set of documents, so that the index can be searched by end-users that want to locate documents that match certain search criteria.

A crawler can retrieve documents from different stores. Although the primary store is the Web, a crawler can retrieve documents from a mail store, database, or any other source that has textual content (the textual content being relevant only for processing of a document for the purpose of indexing, since the crawler itself is not concerned with about what type of document is being crawled).

Currently, in order to search for a product or item on the Internet, a user typically places a search term(s) into a search engine, whereupon the user is provided with a listing of URL's that may or may not contain the desired product. As is typically the result, the search engine results generally include, some products which are not related to the desired product, some products which are tangentially related to the desired product, and many different product models available under a common make, and sometimes the actual desired product, resulting in a collection of listings through which the user must then search. As a result, it is up to the individual user to filter or sort through the search engine results in order to locate the corresponding desired product. As such, the currently available product search techniques for product searching on the Internet yields a large amount of data which is presented to a consumer, wherein the consumer must sort through the data to locate the appropriate desired product.

Further, search engine results are often incomplete or inaccurate where e-commerce sites are concerned for at least two reasons. First, the typical search engines may visit each site as infrequently as once every six months, rendering the data incomplete and / or out of date. For instance, the average time an auction item or product is available on e-Bay is about 2 weeks, as such the search engine would typically miss this type of auction item based upon the

search engine visitation schedule. Second, many e-commerce sites require more than just following links for the crawler to determine what information is available at the particular site.

Moreover, even after the proper product corresponding to the user's search is found from the search results presented, the user has to determine whether the price of the product is a fair or appropriate for that particular product. Presently, a user in not afforded with any readily available mechanism to determine whether the price or valuation of a particular product is high, average, or low, with respect to other identical products, either past or present. As a result, the user is required to research different sources to determine the average range of prices for the same product, in order to determine an appropriate price to pay for the particular product of interest.

Associated with the vast growth of commerce on the Internet is a corresponding consumer burden. A consumer is presented with a vast quantity of information from different service and content providers on the Internet. Thus, a consumer desiring certain types of goods and services has needed an easier way of finding those goods and services from the many different content providers on the Internet. Search engines have generally provided a limited solution to this problem by allowing a consumer to query the Internet using search terms, however, the rapidly emerging Internet business environment still leaves the consumer with too much information, especially for fungible goods. Thus, the consumer must filter through the vast quantity of manufacturers and suppliers offering similar or the same goods and services over the Internet. Some search engines have categorized certain types of goods and services in order to facilitate a search, but the shear volume of merchants, products, and product models remains a difficulty for the average purchasing consumer.

Summary of Invention

A hierarchical product classification system is provided which comprises: logic defining a branched decision tree comprising a plurality of main branches corresponding to different categories of products, a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related

main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; and logic for classifying product entries having product descriptions with a branch terminus of the decision tree by assigning the product entry to a first tree level of the decision tree, and testing the product descriptions against decision node queries leading from the first tree level until a decision node query leading to a branch terminus is satisfied.

An automated method for classifying a product entry within a hierarchical product classification system is provided which comprises: taking a branched decision tree comprising a plurality of main branches corresponding to different categories of products, a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, wherein logic is associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; and classifying a product entry within the decision tree by

assigning the product entry to a first tree level of the decision tree, employing the logic associated with decision node queries to test a product description associated with the product entry to be classified against the decision node queries leading from the first tree level until all of the decision node queries between the first tree level and a branch terminus are satisfied, and classifying the product entry with the branch terminus whose decision node query was satisfied.

According to this embodiment, the first tree level may be manually selected from among the tree levels within the decision tree. The first tree level may be selected via computer executable logic from among the tree levels within the decision tree. The first tree level may also be selected via computer executable logic from among the tree levels within the decision tree with the assistance of a mapping program which correlates the tree levels of the decision tree to a product organization of a location on a network from which the product entry is obtained.

Also according to this embodiment, the computer executable logic may test decision node queries associated with multiple different tree levels in order to select the first tree level.

The first tree level may optionally be randomly selected from among the tree levels within the decision tree.

Optionally, the method may further comprise identifying the product entry to be classified within the decision tree by spidering multiple different sites on a network for the product entry.

The automated method may be repeated for over 500,000 different product entries, for over 1,000,000 different product entries or more.

The automated method may be repeated for multiple different product entries such that at least one branch terminus has at least 100 product entries classified with it, such that at least one branch terminus has at least 500 product entries classified with it, such that at least one branch terminus has at least 1,000 product entries classified with it, such that at least one branch terminus has at least 10,000 product entries classified with it, such that at least one branch terminus has at least 100,000 product entries classified with it or more.

A database is also provided comprising product information aggregated from multiple different sites on a network, the database comprising: computer readable media encoding at least 100,000 different product entries identified from a variety of different sources on a network, each product entry having associated with it information identifying its assignment within a hierarchical product classification system comprising a plurality of main branches corresponding to different categories of products, a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, wherein a given product entry is classified with a branch terminus if a product description of the product entry satisfies the decision node query associated with the branch terminus.

According to one variation of this embodiment, no product entry is classified with more than one branch terminus.

According to one variation of this embodiment, the database comprises at least 1,000,000 different product entries.

According to one variation of this embodiment, at least one product entry in the database is associated with at least 100,000 different branch termini, optionally at least 250,000 different branch termini, optionally at least 500,000 different branch termini or more.

According to one variation of this embodiment, at least 1,000 product entries have been assigned to at least one branch terminus, at least 10,000 product entries have been assigned to at least one branch terminus, at least 100,000 product entries have been assigned to at least one branch terminus or more.

A product search system is also provided which comprises: logic defining a branched decision tree comprising a plurality of main branches corresponding to different categories of products, a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; logic for classifying product entries having product descriptions with a branch terminus of the decision tree by assigning the product entry to a first tree level of the decision tree, and testing the product descriptions against decision node queries leading from the first tree level until a decision node query leading to a branch terminus is satisfied; and logic for searching the classified product entries by matching the decision node queries to a search query, identifying tree level categories and branch terminus categories having decision node queries which match the search query, and returning product categories relating to the identified tree level categories and branch terminus categories.

The product search system may optionally further include returning product entries belonging to the identified tree level categories and branch terminus categories.

The product search system may optionally further include logic for returning product entries relating to one or more product categories selected from among the returned product categories.

An automated method for classifying a product entry within a hierarchical product classification system is also provided which comprises:

taking a branched decision tree comprising a plurality of main branches corresponding to different categories of products, a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, wherein logic is associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; classifying a product entry within the decision tree by assigning the product entry to a first tree level of the decision tree, employing the logic associated with decision node queries to test a product description associated with the product entry to be classified against the decision node queries leading from the first tree level until all of the decision node queries between the first tree level and a branch terminus are satisfied, and classifying the product entry with the branch terminus whose decision node query was satisfied; and searching the classified product entries by having computer executable logic match the decision node queries to a search query, identify tree level categories and branch terminus categories having decision node queries which match the search query, and return product categories relating to the identified tree level categories and branch terminus categories.

The product search method optionally further includes displaying product entries belonging to the identified tree level categories and branch terminus categories.

A system for statistically profiling product information is also provided which comprises: logic defining a branched decision tree comprising a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; logic for classifying product entries having product descriptions with branch termini of the decision tree by assigning the product entries to initial tree levels of the decision tree, and testing the product descriptions against decision node queries leading from the initial tree levels until a decision node query leading to a branch terminus is satisfied; and logic for statistically profiling a group of classified product entries that relate to a selected product subcategory by taking a tree level associated with that product subcategory, identifying product entries classified with branch termini which extend from that tree level, and performing a statistical analysis of at least one property associated with the identified product entries.

According to this system the at least one property is optionally selected from the group consisting of a number of same or similar product entries currently available; a number of same or similar product entries sold; a highest price for product entries sold in that subcategory; a lowest price for product entries sold in that subcategory; an average price for product entries sold in that subcategory; a median price for product entries sold in that subcategory; a first date a product entry in that subcategory was listed on the network; a first date a product entry in that subcategory was sold on the network; a most recent date a product entry in that subcategory was listed on the network; a most recent date a product entry in that subcategory was sold on the network; a frequency that

product entries in that subcategory are listed on the network; a frequency that product entries in that subcategory are sold on the network; and sales price trend information for product entries in that subcategory.

An automated method for statistically profiling product information is also provided which comprises: taking a branched decision tree comprising a plurality of main branches corresponding to different categories of products, a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds, a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends, wherein logic is associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; and classifying product entries within the decision tree by assigning the product entries to a first tree level of the decision tree, employing the logic associated with decision node queries to test a product description associated with the product entries to be classified against the decision node queries leading from the first tree level until all of the decision node queries between the first tree level and a branch terminus are satisfied, and classifying the product entries with the branch terminus whose decision node query was satisfied statistically profiling a group of classified product entries that relate to a selected product subcategory by taking a tree level associated with that product subcategory, identifying product entries classified with branch termini which extend from that tree level, and performing a statistical analysis of at least one property associated with the identified product entries.

According to the method, the at least one property may optionally be selected from the group consisting of a number of same or similar product

entries currently available; a number of same or similar product entries sold; a highest price for product entries sold in that subcategory; a lowest price for product entries sold in that subcategory; an average price for product entries sold in that subcategory; a median price for product entries sold in that subcategory; a first date a product entry in that subcategory was listed on the network; a first date a product entry in that subcategory was sold on the network; a most recent date a product entry in that subcategory was listed on the network; a most recent date a product entry in that subcategory was sold on the network; a frequency that product entries in that subcategory are listed on the network; a frequency that product entries in that subcategory are sold on the network; and sales price trend information for product entries in that subcategory.

According to any of the above, each tree level may optionally comprise at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

Also according to any of the above, a portion of the branches may direct the product description to a different branch when the decision node query associated with the branch is satisfied.

Also according to any of the above, the decision tree may optionally comprise at least 100,000 branch termini, at least 250,000 branch termini, at least 500,000 branch termini or more.

Also according to any of the above, the decision tree may optionally comprise at least 100,000 decision nodes, at least 250,000 decision nodes, at least 500,000 decision nodes or more.

Also according to any of the above, the decision tree may optionally comprise at least 10 tree levels extending between a first tree level and a branch terminus, optionally at least 25 tree levels, optionally at least 50 tree levels, optionally at least 75 tree levels, optionally at least 100 tree levels, or more

Also according to any of the above, at least a portion of the decision node queries specify a negative limitation which if satisfied directs the

classifying logic to exclude the product from the tree level or branch terminus extending from the decision node query.

Also according to any of the above, at least a portion of the decision node queries may optionally be at least 100 words in length, at least 500 words in length, at least 1000 words in length, at least 2000 words in length, at least 3000 words in length, at least 4000 words in length or more.

Brief Description of the Drawings

The present invention is illustrated by way of example in the following drawings in which like references indicate similar elements. The following drawings disclose various embodiments of the present invention for purposes of illustration only and are not intended to limit the scope of the invention.

Figure 1 is a block diagram representing one embodiment of a general purpose computer system in capable of implementing the teachings of the present invention.

Figure 2 is a block diagram representing one embodiment of an exemplary network environment in accordance with one embodiment of the present invention.

Figure 3A represent one embodiment of an aggregated comparative product collection resource for comparing identical products or items that are available from a variety of different resources on a network in accordance with the teachings of the present invention.

Figure 3B represent one embodiment of a valuation profile which may be associated with a product or item in accordance with the teachings of the present invention.

Figure 4A is a block diagram representing one embodiment of a product search and valuation system for consumer product searching capable of implementing the teachings of the present invention.

Figure 4B represents a functional flow diagram illustrating an embodiment of a method for searching a network environment for products or items using the system of the present invention .

Figure 5 provides a highly simplified depiction of the system's taxonomy.

Figure 6 provides a more simplified depiction of the taxonomy tree shown in Figure 5.

Figure 7 illustrates a simplified taxonomy tree which shows in bolded lines the categories which a hypothetical query matches.

Detailed Description of Preferred Embodiments

The following detailed description sets forth numerous specific details to provide a thorough understanding of the invention. However, those of ordinary skill in the art will appreciate that the invention may be practiced without these specific details. In other instances, well-known methods, procedures, protocols, components, algorithms, and circuits have not been described in detail so as not to obscure the invention.

In one embodiment, the steps of the present invention are embodied in machine-executable instructions. The instructions can be used to cause a general-purpose or special-purpose processor that is programmed with the instructions to perform the steps of the present invention. Alternatively, the steps of the present invention might be performed by specific hardware components that contain hardwired logic for performing the steps, or by any combination of programmed computer components and custom hardware components.

The present invention is generally directed to a method and system that provides a network- based service, preferably Internet-based, for consumer product and services searching. The network based service provides a consumer with a comparative resource of identical products or services available on a network, such as the Internet, as well as, valuation information associated with each available corresponding product or service.

1. Computer Environment

Figure 1 and the following description are intended to provide a general description of a suitable computing environment in which the invention may be implemented. Although not necessarily required, the invention will be described in the general context of computer-executable instructions, such as program modules, being executed by a computer, such as a client workstation or a server. Generally, program modules include routines, programs, objects, components,

data structures and the like that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, including hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers and the like. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

As shown in Figure 1, an exemplary general purpose computing system may include a conventional personal computer 20 or the like, including a processing unit 21, a system memory 22, and a system bus 23 that couples various system components including the system memory 22 to the processing unit 21. The system bus 23 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory 22 may include read-only memory (ROM) 24 and random access memory (RAM) 25. A basic input/output system 26 (BIOS), containing the basic routines that help to transfer information between elements within the personal computer 20, such as during start-up, may be stored in ROM 24. The personal computer 20 may further include a hard disk drive 27 for reading from and writing to a hard disk (not shown), a magnetic disk drive 28 for reading from or writing to a removable magnetic disk 29, and an optical disk drive 30 for reading from or writing to a removable optical disk 31 such as a CD-ROM or other optical media. The hard disk drive 27, magnetic disk drive 28, and optical disk drive 30 may be connected to the system bus 23 by a hard disk drive interface 32, a magnetic disk drive interface 33, and an optical drive interface 34, respectively. The drives and their associated computer-readable media provide non-volatile storage of computer readable instructions, data structures, program modules and other data for the personal computer 20. Although the exemplary embodiment described herein may employ a hard disk, a removable magnetic disk 29, and a

removable optical disk 31, or combination therefor, it should be appreciated by those skilled in the art that other types of computer readable media which can store data that is accessible by a computer, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, random access memories (RAMs), read-only memories (ROMs) and the like may also be used in the exemplary operating environment.

A number of program modules may be stored on the hard disk, magnetic disk 29, optical disk 31, ROM 24 or RAM 25, including an operating system 35, one or more application programs 36, other program modules 37 and program data 38. A user may enter commands and information into the personal computer 20 through input devices such as a keyboard 40 and pointing device 42. Other input devices (not shown) may include a microphone, joystick, game pad, satellite disk, scanner, or the like. These and other input devices are often connected to the processing unit 21 through a serial port interface 46 that is coupled to the system bus 23, but may be connected by other interfaces, such as a parallel port, game port, or universal serial bus (USB). A monitor 47 or other type of display device may also be connected to the system bus 23 via an interface, such as a video adapter 48. In addition to the monitor 47, personal computers may typically include other peripheral output devices (not shown), such as speakers and printers.

The personal computer 20 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 49. The remote computer 49 may be another personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the personal computer 20, although only a memory storage device 50 has been illustrated in Figure 1. The logical connections depicted in Figure 1 include a local area network (LAN) 51 and a wide area network (WAN) 52. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, and the Internet.

When used in a LAN networking environment, the personal computer 20 is connected to the LAN 51 through a network interface or adapter 53. When

used in a WAN networking environment, the personal computer 20 typically includes a modem 54 or other means for establishing communications over the wide area network 52, such as the Internet. The modem 54, which may be internal or external, is connected to the system bus 23 via the serial port interface 46. In a networked environment, program modules depicted relative to the personal computer 20, or portions thereof, may be stored in the remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

2. Network Environment

As noted, the computer described above can be deployed as part of a computer network. In general, the above description applies to both server computers and client computers deployed in a network environment. Figure 2 illustrates one such exemplary network environment in which the present invention may be employed. As shown in Figure 2, a number of servers 10a, 10b, etc., are interconnected via a communications network 160 (which may be a LAN, WAN, intranet or the Internet) with a number of client computers 20a, 20b, 20c, etc. In a network environment in which the communications network 160 is, e.g., the Internet, the servers 10 can be Web servers with which the clients 20 communicate via any of a number of known protocols such as, for instance, hypertext transfer protocol (HTTP). Each client computer 20 can be equipped with a browser 180 to gain access to the servers 10, and client application software 185. As shown in the embodiment of Figure 2, the server 10a may include or be associated with a dynamic database 12.

As shown, the database 12 may include database fields 12a, which contain information about items stored in the database 12. For instance, the database fields 12a can be structured in the database in a variety of ways. The fields 12a could be structured using linked lists, multi-dimensional data arrays, hash tables, or the like. This is generally a design choice based on ease of implementation, amount of free memory, the characteristics of the data to be stored, whether the database is likely to be written to frequently or instead is

likely to be mostly read from, and the like. A generic field 12a is depicted on the left side. As shown, a field generally has sub-fields that contain various types of information associated with the field, such as an ID or header sub-field, type of item sub-field, sub-fields containing characteristics, and so on. These database fields 12a are shown for illustrative purposes only, and as mentioned, the particular implementation of data storage in a database can vary widely according to preference.

Thus, the present invention can be utilized in a computer network environment having client computers for accessing and interacting with the network and a server computer for interacting with client computers and communicating with a database with stored inventory fields. Likewise, the consumer rewarding process of the present invention can be implemented with a variety of network-based architectures, and thus should not be limited to the examples shown. The present invention will now be described in more detail with reference to preferred embodiments.

3. Product Search, Classification and Statistical Profiling Architecture

The system of the present invention searches for products associated with a network and organizes those products into a series of product or item groups (e.g., categories or category groups), illustrated in Figure 3A, such that the system provides one or more aggregated product collections. These product collections serve as a comparative resource for comparing groups of identical products and categories of similar products that are available from a variety of different resources on a network, such as the Internet. The system also serves as a historical resource for the identified products in the sense that the system enables statistical profiling of the products and categories of similar products which is based on data currently available on the network as well as data which was previously available on the network.

The term 'product' or 'item', as it is used herein, may represent any type of product, service, information, or data, and is used interchangeably throughout

this description. The term 'identical product' or 'identical item' is used to identify products or items that possess the same descriptive attributes or identification profile, and is used interchangeably throughout this description.

In one embodiment, the system of the present invention is configured to provide a statistical profile, illustrated in Figure 3B as a valuation profile, which may be associated with a desired product or item maintained by the system and viewed by the user / consumer. This allows a consumer interested in a particular product to compare the desired product or item price to a generated average valuation profile associated with the particular product or item.

Data accumulated by the system allows the system to create a variety of profiles for a product or item other than valuation profiles. For example, the below table summarizes some of the different statistical profiles that may be created for a particular product or category of products by the system.

Examples of Product or Category Profiles

(a) Number of same or similar items currently available
(b) Number of same or similar items sold
(c) Highest price of items that sold in that category;
(d) Lowest price of items that sold in that category;
(e) Average price of items that sold in that category;
(f) Date first item was listed in that category;
(g) Date first item was sold in that category;
(h) Most recent date an item was posted in that category;
(i) Most recent date an item sold in that category;
(j) Frequency with which items are posted in that category;
(k) Frequency with which items are sold in that category;
(l) Sales price trend information for items in that category to help users determine whether the price an item sells for is increasing or decreasing over time; and
(m) "Top XX Lists" that show the XX number highest-ranking items for a different statistical measure, such as those described above.

The above table is not meant to be exhaustive but rather reflects the broad range of statistical profiles that may be created using the system of the present invention.

As described herein in greater detail, the system employs a comprehensive hierarchical product classification system, referred to herein as the system's "Taxonomy" which enables the diverse range of products identified by the system to be finely categorized. The Taxonomy comprises a tree structure of product categories and subcategories. Products that are identified by a search are classified within the various categories and subcategories within which the product belongs.

The system may also provide a series of statistical profiles for a product or item (such as those described above) for each of the product categories to which the product is assigned in the Taxonomy. As such, the system provides an in-depth comparative resource for comparing a particular product to other products or classes of products currently and/or previously identified by the system of the present invention.

The system is also configured to collect a variety of item or product data characterizing and corresponding to a series of products or items that are available on the network (e.g., Internet). Accordingly, the product or item data (e.g., description of product or item) and a corresponding different profiles for the particular product or item, may be presented to a user through an interface, which allows the user to purchase, use, consider, or otherwise compare products or items available on the network to each other.

Data about the products or items included in the system of the present invention may be acquired from a variety of different sources, such as, but not limited to, e-auction sites, electronic classified listings, merchants with e-commerce sites, specific URL locations, and a variety of other product source sites. As such, the user is presented with a collection of identical products that are available from a variety of different sources, thereby allowing the user to compare the items or products to each other.

For instance, the user may select a product such as a gold PRESIDENT ROLEX™ watch, whereupon the user is then presented with a collection of

identical products or items (i.e., gold PRESIDENT ROLEX™ watches) which may be available from a variety of different sources (e.g., e-auction sites, electronic classified listings, merchants with e-commerce sites, specific URL locations, and a variety of other product source sites). Accordingly, the user may then review prices, product conditions, and any other information associated with each identical product (i.e., gold PRESIDENT ROLEX™ watches) in order to determine which gold PRESIDENT ROLEX™ watch represents the best value.

Figure 4A illustrates one embodiment of a product search and valuation system 400 according to the present invention, in block-flow diagram format, that provides a network-based service, preferably Internet-based, for consumer product searching. Figure 4B represents a functional flow diagram illustrating an embodiment of a method for searching a network environment for products or items using the system of the present invention .

The system of the present invention and associated method provides a consumer with a collection of products offered on a network, such as the Internet, that are available for purchase, use, comparison, or consideration by the consumer.

As illustrated in Figure 4A, the system 400 comprises a spider module 405, ItemData table 410, a classification module (classifier) 415, a Taxonomy 420, an AllItems table 425, and a profiling module 430, which are configured to be in selective communication with each other, as necessary or as desired, to perform in accordance with the following description. It is understood that different components of the system of the present invention may be omitted or added to the system, as desired, to provide for different functionality models or operation.

Figure 4B illustrates in flow chart format the system's process for searching a network environment, such as the Internet, for products or items, and classifying those products. Initially, at searching step 460, the system searches the network for information regarding products that are available for purchase, use, comparison, or consideration. This step is performed by the spider module 405. The product or item data may be acquired from a variety of

different sources, such as, but not limited to, e-auction sites, electronic classified listings, merchants with e-commerce sites, specific URL locations, and a variety of other product source sites. The item information identified by the searching step 460 is aggregated in ItemData records and stored in ItemData table 410.

The aggregated records are then classified during classification step 465 by a classification module (classifier) 415. The classification step 465 is performed to determine the proper classification for each item identified by the search within the system's taxonomy 420. As will be described herein in greater detail, the system's taxonomy 420 is a comprehensive hierarchical product classification system which finely categorizes the diverse range of products identified by the system.

Once an item or product has been classified (e.g., classified ItemData) by being assigned to different portions of the system's taxonomy 420, the item is marked as having been processed and a corresponding entry for the ItemData is made in the AllItems Table 425, at ItemData entry step 470A. As illustrated in 470B, in some instances, an item or product (e.g., ItemData record) cannot be assigned to a category of the system's taxonomy. In such instances, the item is not marked as having been processed, and no entry is made for the item in the AllItems table. In this way, users may not be able to view or otherwise see unclassified ItemData record. In an alternate embodiment, illustrated in ItemData assignment step 470C, if an item or product (e.g., ItemData record) could not be assigned to a taxonomy category a containership "walk back", looking for the first (and closest) category in which to put the item or product is performed. This virtually guarantees each item will receive at least some classification within the system's taxonomy. If too many items have difficulty being classified, the content editors may devise a classification arrangement to address why the item or product needed to undergo ItemData assignment step 470C.

The system of the present invention also provides statistical profiling of items, at profiling step 475 by profiling module 430. Profiling of items may be associated with a desired or selected product or product category. As will be

explained herein in greater detail, the profile may be related to price and may also be related to a wide variety of other statistical values or trend information derived from the body of products classified within the system's taxonomy. The profiling step is configured to provide a statistical profile for a desired or selected item or product, based upon information accumulated for that corresponding item (e.g., accumulated classified ItemData). As also explained herein, the profiling step 475 may also be used to analyze whether the system's taxonomy needs to be modified.

Each of the above described modules and steps will now be described in greater detail.

A. Spider Module 405

As illustrated in the embodiment of Figure 4, the system 400 includes a spider module 405 in operative communication with a network, such as the Internet, which allows communication of information, via any of a number of known protocols such as, for instance, hypertext transfer protocol (HTTP), with other associated networks and server devices. The spider module 405 may be deployed as part of a computer network or an individual computing device. In general, the above description applies to both server computers and client computers deployed in a network environment.

Accordingly, the spider module 405 is configured to search the Internet or other network environment for different products or items based upon a series of product or item descriptors, which may be supplied from an item descriptor table, described in further detail below. In one embodiment, the product or item descriptors are maintained in the system's taxonomy 420.

In one embodiment, the spider module 405 comprises a site spider module 405A, a spider registry module 405B, and a spider agent module 405C. Accordingly, in one embodiment, the site spider 405A comprises a server program for example, that understands the details of how a particular site, such as YAHOO™ for example, operates and is organized. Likewise, in one embodiment, the spider registry 405B is a central clearinghouse server for

example, through which the site spider modules 405A request that the spider agents 405C retrieve product or item data, web pages, or any other data, from a particular site. The spider agent 405C, in one embodiment, is a slave program for example, that is configured or enabled to visit different sites (i.e., Internet websites) and fetch and process their respective contents as directed by the site spider 405A.

As such, the site spider 405A operates to retrieve the available product or item information (e.g., available auction products, available classified products, etc.) offered at a particular site, also referred to as an ItemData or ItemData records, as the site spider 405A is configured to understand the details of how to access the information or content of a particular site. At any particular instance, multiple site spiders 405A may be running or otherwise be enabled. The site spider 405A sends requests for tasks (e.g., load a list of top-level auction categories, product search tasks, etc.) to the spider registry 405B. It is the responsibility of the individual spider agents 405C, of which there may be many running at any particular instance, to retrieve this information from a particular site. In one embodiment, there is no context or history stored by a spider agent 405C between tasks; tasks are assigned to a spider agent 405C without regard to the task that a spider agent 405C may have handled in the past. For example, the spider agent 405C may process a task for the YAHOO™ auction site one moment and be given an AMAZON™ product task next. Correspondingly, the spider agents 405C may operate remotely over the Internet, and the spider registry 405B is capable of managing any number of them.

The types of products or items that are retrieved from the network by the spider module 405 are based upon product or item descriptors, which are supplied to the spider module 405 from the spider registry 405B. Accordingly, in one embodiment, the retrieved products or items, referred to as ItemData, are then placed or temporarily stored in an ItemData table 410.

In one embodiment, the product or item descriptors, which are used to search for corresponding products or items, are generated by personnel associated with the system 400. In an alternate embodiment, the spider registry

405B obtains the product or item descriptors from decision node queries of the system's taxonomy 420.

As such, the spider module 405 uses the product or item descriptors as the 'search' criteria for locating the corresponding products or items on the network (e.g., Internet). Correspondingly, the product or item descriptors may be configured into a collection of different product or item descriptors categorized into a series of different product categories. As such, each product category contains a collection of products (e.g., product descriptors) which corresponds to a particular product category. The product or item descriptors provide the descriptive attributes or identification profile of the products or items that are to be searched for in the network.

In one embodiment, the series of product or item descriptors may be broken down or decomposed into a series of subcategory descriptors. The subcategory descriptors may be further broken down into micro-subcategories. As such, each product or item descriptor may be broken down into varying nth degrees of subcategories and micro-subcategories, and so on, as desired or required in order to accurately describe an individual product or item.

For example, a product or item descriptor may contain the term 'jewelry', which may also comprise the product or item category. The product or item descriptor subcategory may then comprise the term 'watches', and the 1st degree micro-subcategory may comprise the term 'ROLEX'™, the 2nd degree micro-subcategory may comprise the term 'gold', and the 3rd degree micro-subcategory may comprise the term 'PRESIDENT', and so on. As such, for the above example, the product or item descriptor would be as follows: Jewelry: Watch: ROLEX™: Gold: PRESIDENT; which would describe a gold PRESIDENT ROLEX™ watch within the system's taxonomy.

In one embodiment, the spider module 405 uses the product or item descriptors as the 'search' criteria for a corresponding product or item in the network. Additionally, the product or item descriptors may include a variety of different variables, including, but not limited to, price variables or price ranges, age variables, condition variables (e.g., new or used), availability variables (e.g., 1 of 5 available), and a variety of other search variables which may be used, as

desired, by the spider module in locating corresponding identical products or items. Moreover, the product or items descriptors may be edited, added to, or removed, as desired to perform according to a desired protocol.

In one embodiment, the spider modules 405 blindly adds Item Data records to ItemData table with no other context than what the Spider Registry 405B tells them, and at some point in time, the classification module 415 will query a Sites table. The classification module 415 then determines which sites have been marked as "suitable for classification" and will pull those records (e.g., ItemData records) from ItemData table 410, map them, and thereby move them into an AllItems table 425, without once having exchanged information with any spiders.

In an alternate embodiment, a user may simply input a search term or string of search terms (via search interface) into the spider module 405, whereupon the spider module 405 then searches the Internet to locate the corresponding products or items that corresponds to the search term or string of search terms input by the user.

In one embodiment, the spider module 405 is configured to collect the following product or item information, in whole or part, that may be associated with a product or item on a particular site: (1) NAME: The name or title used to refer to a particular product or item; (2) DESCRIPTION: The description of the product or item, or how the product or item is being described; (3) SITE INFORMATION: The location or source of the product or item, URL; (4) PRICE / BID INFORMATION: Asking price, number of bids, range of bids (high to low range), and any other pricing information associated with a product or item (e.g., barter value, etc.); (5) CLOSE DATE: Expected time the product or item is to become unavailable or closing time of auction for the product or item; (6) TAXONOMY: Where in the source site was the product or item filed or otherwise maintained (i.e., the taxonomy at the site of origin) and where in the system is the product or item information to be filed or maintained (i.e., the taxonomy of the system, discussed in further detail below); (7) CLOSURE / WITHDRAWAL: The spider module 405 monitors different sites (e.g., auctions, classifieds, etc.) to determine closure on a product (e.g., the auction

has ended) or withdrawal (e.g., the product or item was withdrawn from availability), the spider module 405 communicates this information to the system of the present invention 400 such that products or items which are no longer available are not displayed to a user or consumer; (8) UPDATES: The spider module 405 communicates changes associated with the product or item (e.g., price or description changes) to the system 400 as the changes are encountered; (9) SELLER: The identifier (e.g., URL, name, etc.) of the individual or merchant offering the product or item for sale; and (10) SITE TYPE: the type of site, such as, auction sites, retail sites, classified ads sites, EBAY™, or a variety of other sites, such that the spider module associates the site type with product or item.

In one embodiment, the ItemData retrieved by a spider agent 405C is processed upon retrieval and is not stored to or on any permanent medium, which provides increased compression and data flow efficiency. This is accomplished through a technique, referred to herein as babelizing, wherein the spider agent 405C is configured to strip information deemed to be non-essential from the text of the ItemData retrieved (such as words that do not further describe the item, e.g., and, the, or). By babelizing the ItemData, a condensed version of the ItemData is created because it contains fewer characters. In one embodiment, the babelized ItemData may be a collection or list of words from the original text of the ItemData, rather than the actual text itself.

In one embodiment, the spider module 405 may be configured to search for products or items at specified intervals in order to maintain the categorized collection of classified ItemData in the AllItems table 425. It is important, however, that the spider module 405 be able to tell quickly whether they have already seen a particular product or item or whether it has not. Generally, most sites allow access to summary information listing a number of products of items on a single web page, as well as, a detail view that shows a single product or item on one or more web pages.

In one embodiment, the spider module 405 is configured to access the full description twice for efficiency purposes, once upon first encountering the product or item and once when the product or items closes or is withdrawn from

availability. Accordingly, in one embodiment, the spider module 405 is configured to maintain an ActiveItems table that serves to identify which products or items have already been retrieved from a particular site. Further, the ActiveItems table may be configured to contain a key field called *SiteItemID*, which contains site-specific information that uniquely identifies the product or item.

The site spider 405A contains the high-level logic necessary to traverse a particular target site, for example, the AMAZON™ auction site. In one embodiment, the site spider 405A operates by performing two information queries or 'passes' on a particular site, the first query or 'pass' locates all products or items that are currently open or available on a site by visiting the site recursively, beginning with a top-level category page. A category page contains either subcategories or leaf categories. A leaf category contains one or more result lists. A result list contains a series of item URLs (from which an Item ID can be extracted), current price, or other information for that product. Optionally, there may be a Next Link in a result list. The Item IDs and prices may be stored in a table for later processing. The Next Link is processed to fetch the next series of Item IDs. Once a result page for a given category is fetched that has no Next link, that category is complete.

Once all leaf categories have been processed, Delta Processing takes place. Delta Processing determines which items or products are new to the site by comparing the list of open products (e.g., available products, such as contained in the AllItems table 425) to those that were open on the site spider's most recent visit to the site. Similarly, Delta Processing determines which items have closed (e.g., non-available product) since the site spider module's last visit. Products or items are assumed to have been Closed if they were Open in the most recent run, but not found in the current run. The Delta processing may be applied to e-auction sites, electronic classified listings, merchants with e-commerce sites, specific URL locations, and a variety of other product source sites.

At the end of delta processing, an output table is constructed for the second information query or 'pass'. The output table may comprise a list of

New Items, Closed Items, and Update items. The second query or 'pass' works through the table created by the first query or 'pass', handling the New, Closed, and Update records that the output table contains. New items or products are processed by requesting a Get operation from the spider registry 405C (and the associated Spider Agent 405A that actually performs the operation). When the task is complete, the columns of parsed product or item information is written to the ItemData table 410, indicating that a new item or product has been found. Simultaneously, the site spider 405A notes the new item in its private ActiveItems table, so that subsequent runs will recognize that this item has been found on a previous run.

Closed items are handled by requesting a GetClosed task from the spider registry 405B. The GetClosed task processing performed by a spider agent 405C checks to see whether an item or product actually sold, and if so, the price it sold at. Accordingly, the price at which an item or product sold is associated with the product for later reference. By tracking the prices at which products sold, pricing data for the product and the class and subclasses to which the product belongs can be compiled. As will be described herein in greater detail, the price information may be sent to the profiling module 430 and may be used by the profiling module 430 to construct a valuation profile for any other identical products to which the sold product or item correlates to, as well as hierarchical categories to which the product may belong. For example, the price of a gold PRESIDENT ROLEX™ watch may be associated with the following categories: gold PRESIDENT ROLEX™ watches; gold ROLEX™ watches; ROLEX™ watches; and watches.

Update items typically contain only the most recent price information. Accordingly, the items and products that are closed or are otherwise no longer available are generally removed from the AllItems table 425, so as not to be visible to a user or consumer. The AllItems table 425 comprises a collection of products that are available for purchase, use, comparison, or consideration by a consumer.

The spider registry 405B serves as the point of contact for site spiders 405A and the multiple remote spider agents 405C that will perform the tasks

required by a particular site spider 405A. There is generally a single spider registry instance in a given installation, although this is not a requirement. Site spiders 405A use the spider registry 405B to perform tasks; generally, the tasks involve reading and parsing a web page. To do this, site spiders 405A call a DoTask() interface, passing in task-specific arguments. In one embodiment, the DoTask() blocks the caller until the task has been completed.

Spider Agent (many) ←----------→ Spider Registry ←----------→ Site Spider (many)

Spider agents 405A (typically, more than one) connect with a spider registry 405B and loop continuously, fetching tasks, executing tasks, and reporting task results. In one embodiment, the spider registry 405B contains logic to automatically retry tasks if results are not returned within a predefined time period, as Internet connections are generally not 100% reliable. Remote spider agents 405C connect and announce their availability for task processing via Register() and Unregister() calls. They use the GetTask() and ReportTaskResult() functions to fetch tasks and report their results, respectively. In one embodiment, the spider agents 405C may be configured to run on any Win32 machine that is connected to the Internet or other network.

In one embodiment, the spider agent 405C consists of three components: A Master Dialog; an Agent; and a Task Processor. The Master Dialog controls the operation of the spider agent 405C as a whole, and contains UI components to Stop, Start, and Exit. Additionally, there are edit fields for specifying which spider registry server 405B (local network machine name or IP address) with which the Agent should converse (i.e., get tasks from and report task results to). In one embodiment, the Agent is a unit of software that actually conducts the FetchTask/ ExecuteTask/ Report Results protocol with the spider registry 405B. Once started, the Master Dialog is configured to create multiple independent Agents. In one embodiment, the Master Dialog shuts down Agents when the UI directs. The Task Processor is a unit of software actually able to carry out tasks that are given to Agents by the spider registry 405B. It contains specific knowledge of the tasks that each site spider 405A may need to accomplish.

B. ItemData Table 410

As illustrated in Figure 4, the spider module 405 is operatively associated with a ItemData table 410. The ItemData table 410 comprises a collection of raw product or item information, also referred to as ItemData, ItemData records, or unclassified ItemData, retrieved by the spider module 405. In one embodiment, the ItemData table 410 is configured as a storage medium for storing the ItemData retrieved from the Internet by the spider module 405. In an alternate embodiment, the ItemData table 410 is configured to operate as a data conduit, wherein the ItemData accumulated or retrieved by the spider module 405 is passed onto a subsequent process or application by the ItemData table 410. The spider module 405 adds ItemData to the ItemData table 410 as the products or items are encountered in the network.

As mentioned above, the spider module 405, using the site spiders 405A, feeds ItemData into the ItemData table 410. Each spider uses its own techniques to walk a web site's item or product categorization and to add ItemData to the ItemData table 410 for every product or item the spider 405 finds. Each ItemData record captures the following important information, in part or in whole, about each product or item:

- SiteID: the integer value associated with this site in the ***Sites*** table.

- Category the site-relative category "path" (e.g. "Auctions > Antiques > Pre-1900. . .")

- Name the short description for an item

Description the long description for an item

ItemDataID Unique PK identifier for this item

- SiteItemID Free-form unique identifier for this item (which spider module knows how to interpret)

- URL URL to this item on specific auction site

- Seller	Seller's name as shown on site
- StartDate	Auction start date for item
- CloseDate	Date when item is expected to close
Auction Type	EBAY™/auction/classified/retail/etc.
ItemData Type	Small status field to indicate open/update/close

The spider module 405 adds the ItemData records to the ItemData table 410 as the products or items are encountered, and the classification module 415 reads ItemData records out of ItemData table 410 asynchronously with respect to the spider module 405. As such, the spider module 405 and classification module 415 operate independently of each other. Further, in an alternate embodiment, the classification module 415 may read the ItemData records out of ItemData table 410 synchronously with respect to the spider module 405, if such is desired.

In one embodiment, the ItemData table 410 can be thought of as a large buffer of products or items (ItemData records) that have been collected by spider module 405 but not yet processed by the system 400. It is noted that a variety of systems, other than the one described above with regard to the spider module, may be used to accumulate product data that is to be stored in the ItemData records. These other systems may be then used with the remainder of the system of the present invention.

In one embodiment, the spider module 405 is configured to maintain a ***Sites*** table identifying the respective sites (SiteID) that the spider module 405 has visited. In this regard, additional sites may be added to the ***Sites*** table as additional sites are identified.

In one embodiment, for efficiency, the ItemData records for each site are processed together (i.e. all YAHOO items are processed, then all EBAY items, etc.) This reduces the query overhead of joining the Sites and the ItemData table 410.

C. Taxonomy 420

One of the significant challenges addressed by the system of the present invention is the automated classification by the system 400 of the multitude of different products identified by the spider module 405 that are stored in the ItemData records. When information about products is aggregated from different websites, the information is initially in a format that is based on the site from which the information was obtained. Since information needs to be accessed and retrieved from multiple different sites, the product information that is retrieved is in multiple different formats. A need thus exists to place all of the product information into a common format.

A further classification challenge is created by the fact that the same or similar products are frequently not described in a consistent manner. This is particularly true for websites, such as EBAY, where multiple individual users post their own information. Inconsistencies in how the same or similar products are described makes the automated classification of products very difficult.

The Taxonomy 420 employed by the system of the present invention is used to help overcome these challenges in classifying products based on information obtained from different websites by providing a hierarchical product classification system by which different products identified by the spider module 405 and stored in ItemData records of the ItemData table 410 may be classified. The classification module, 415, in combination with the Taxonomy 420, provides mechanisms for performing the classifications which overcome existing problems associated with products having varying descriptions and description formats on the web.

The classification module 415 functions to review information contained in the ItemData table 410, and determine the proper product category for each

corresponding classifiable ItemData within the Taxonomy 420. Once the classification module 415 assigns an item from the ItemData table to a category within the system's taxonomy 420, the item is added to the AllItems table 425, along with the item's taxonomy assignment. The AllItems table 425 comprises a collection of products or items (i.e., classified ItemData records) which are available for purchase, use, comparison, or consideration by a consumer. The classified product records placed in an AllItems table 425 may be accessed by a user of the system 400.

ItemData records which prove not to be classifiable within the system's taxonomy 420 may be left in the ItemData table 410, where they can be reprocessed at a later time (presumably after a content editor has noticed the unclassified ItemData records accumulating, wherein the content editor may modify the system's taxonomy to allow the records to be classified).

In one embodiment, the spider module 405 analyzes the ItemData, such as by analyzing a small status field which tells content editors what is "wrong" with the raw data or ItemData record, to determine the reason that the ItemData is not classifiable.

Figure 5 provides a highly simplified depiction of the system's taxonomy. As can be seen, the taxonomy provides a hierarchical product classification system comprising a branched decision tree 510 with a series of tree levels 515A-E and branches 520A-E extending from each tree level. Decision nodes 525 are positioned at the intersections leading to each sub-branch 520A-E of the decision tree 510. As one goes down the decision tree 510, the branches narrow and ultimately terminate in a series of branch termini 530, also referred to herein as a leaf.

Each tree level 515 in the taxonomy relates to a product category that is defined by the decision node 525 above the tree level 515. A query associated with the decision node 525 above the particular tree level 515 needs to be satisfied in order for a given product to be associated with that tree level 515. As such, the decision node query serves as a filter to exclude products that do not fit within the product category that the tree level is designed to represent. The branches 520 extending from a given tree level 515 provide possibilities as

to how that product category may be further subdivided. As illustrated, some branches 520 extending from a given tree level 515 lead to a branch terminus 530 or leaf. Meanwhile, other branches extend to additional tree levels.

The queries associated with the decision nodes 525 leading to each branch represent a way of testing a given product description with regard to whether it should be associated with the further subdivision represented by the branch 520 below the decision node 525. The branch termini 530 relate to a category that a product is assigned to when the branch has no further sub-branches. A branch terminus 530 may be associated with each tree level 515 to provide a category to assign a product description to when it does not satisfy any of the other decision nodes 525 at that tree level 515. By reviewing what products accumulate in a given branch terminus, an editor using a Taxonomy Administration Tool can modify the system's taxonomy to include further tree levels, branches, decision nodes and branch termini that serve to further subdivide the product categories.

The design and operation of the system's taxonomy may be further understood via the simplified example provided in Figure 6. Figure 6 provides an even more simplified depiction of the system's taxonomy as it relates to the assignment of the following three product descriptions: (a) "ROLEX Gold DAYTONA watch"; "PRESIDENT's ROLEX watch, 18k gold"; and (c) "gold ROLEX watch." As can be seen, decision node 610A on the top tree level shown 615A sets forth a query of whether the product description includes the term "watch." Each product description contains that term and thus satisfies decision node 610A.

Decision nodes on the second tree level shown 615B sets forth queries regarding the brand of watch. Decision node 610B query specifies "rolex?." Each product description contains the term "rolex" and thus satisfies decision node 610B.

Decision nodes on the third tree level shown 615C set forth queries regarding the type of ROLEX watch. Decision node 610C query specifies "gold or DAYTONA or PRESIDENT, a way of querying for different types of

gold ROLEX watches. Since each product description satisfies decision node 610C, each product description proceeds to the fourth tree level 615D.

The fourth tree level 615D includes decision node queries for PRESIDENT 610D, DAYTONA 610E, and "no node satisfied" 610F. A product having the description "ROLEX Gold DAYTONA watch" satisfies decision node 610E and becomes associated with branch terminus 630A. Because the branch below decision node 610E is a terminus, this product description is not queried further. A product having the description "PRESIDENT's ROLEX watch, 18k gold" satisfies decision node 610D and is associated with branch terminus 630A. Again, because the branch below decision node 610D is a terminus, this product description is not queried further. Meanwhile, a product having the description "gold ROLEX watch" does not satisfy any of the decision node queries other than decision node 610F, and thus is associated with branch terminus 630C.

It is noted that each product description is not only associated with the decision node 610 controlling the branch terminus 530 with which the product description is associated. The product description is also associated with each decision node 610 that the product must have satisfied in order to reach that branch terminus. Hence, product description "ROLEX Gold DAYTONA watch" is associated with decision nodes 610A, 610B, and 610E. Meanwhile, "PRESIDENT's ROLEX watch, 18k gold" is associated with decision nodes 610A, 610B, and 610D and "gold ROLEX watch" is associated with decision nodes 610A, 610B, and 610F.

The association of product descriptions with the set of decision nodes that the product description satisfies (or must satisfy) in order to be associated with a particular branch terminus is used to facilitate searching for products and characterizing product groups, as described herein in greater detail.

Referring back to Figures 5 and 6, the squiggly lines 535 are used to signify that the portions of the decision trees shown in these figures are but a small fraction of the overall taxonomy decision tree. It is noted that the system's taxonomy is designed to accommodate a very large and diverse group of products, more specifically any product or service which may now or in the

future be advertised or offered for sale on the web. As a result, the system's taxonomy preferably has over 100,000 branch termini or leafs, more preferably over 250,000 and most preferably over 500,000.

The overall taxonomy is designed to accommodate millions of individual product entries. Accordingly, each branch termini or leaf may contain multiple individual product entries (e.g., >100, > 1000, >10,000 and optionally >100,000) for that highly divided subcategory. For this reason, each branch termini is depicted as a database in Figures 5 and 6 within which these multiple individual product entries may be stored or related.

Given the extremely large number of product descriptions to be classified, and the large number of classifications that are provided by the system's taxonomy, a very large number of decision nodes are employed in the decision tree. For example, the system's taxonomy preferably has over 100,000 decision nodes, more preferably over 250,000 and most preferably over 500,000.

Some branch termini require satisfying only one decision node. However, more typically, multiple decision nodes need to be satisfied in order to reach a given branch termini. For example, Figure 6 shows four decision nodes being satisfied in order to assign any of the three product descriptions. This is a highly simplified example. In some instances, the system's taxonomy can require satisfying > 10, >25, >50, >75, > 100, > 125 and even >150 decision nodes in order to reach a particular branch terminus. In one instance, 180 decision nodes need to be satisfied in order to reach a particular branch terminus. Taxonomies may be designed where an even greater number of decision nodes need to be satisfied in order to reach particular branch terminus.

Figure 6 illustrates several simplistic decision node queries. It is noted that far more complex decision node queries are actually required in order to capture all of the possible ways that someone might draft a product description to describe a particular product category. For example, in order to identify all product descriptions relating to watches, one needs a more complex query than ="watch"?. For example, the query would need include terms to cover the singular and plural forms of "watch", synonyms of "watch", misspellings of

"watch" and its synonyms, brands of watches, and models of watches. One might also want to incorporate negative queries, to avoid misclassifications. For example, one might design a query to exclude "watch out for hidden shipping costs from our competitors."

According to the present invention, queries used in conjunction with the decision node queries may be >100 words long, optionally >500 words long. In some instances, the queries may be >1000 words, >2000 words, >3000 words and even >4000 words in length. In one instance, a query of over 8000 words has been used. Longer queries may also be used.

It is noted that some products may be classified in multiple disparate categories. For example, binoculars may be classified under bird watching, sporting event equipment, and opera. Accordingly, although the branched decision tree shown in Figures 5 and 6 is illustrated without any cross branching, it is recognized that it may be desirable to cross-link certain branches of the system's taxonomy or forward a product to selected branches of the system's taxonomy in view of these category overlaps.

D. Classification Module 415

The classification module performs the process of assigning an item or product (e.g. ItemData) a *Taxonomy ID* to indicate the proper location of the classified ItemData record within the system's taxonomy 425. As described with regard to Figures 5 and 6, assigning a *Taxonomy ID* for a classified ItemData record involves identifying which branch terminus or leaf is suitable for a given product description. By identifying which branch terminus is suitable for a given product description, the system also inherently identifies the list of decision nodes which the product description also satisfies. For example, with regard to Figure 6, a product description assigned to branch terminus 630C inherently identifies that the product description satisfies decision nodes 610A, 610B, 610C and 610F.

Classification is a particularly critical process since the system's taxonomy serves as the system's knowledge base about classified ItemData

records. More specifically, the system uses the assigned branch terminus for a given product and list of satisfied decision nodes, as identified by the assigned *Taxonomy ID*, to search for and analyze product listings stored in the AllItems table.

In essence, all information that the system of the present invention has about a classified ItemData record (aside from what was in its original listing) is derived from how the ItemData record is assigned within the system's taxonomy 425. This is because the branch terminus to which a product listing is assigned in the system's taxonomy corresponds to what retailers call SKU's, or Stock Keeping Units. A SKU (pronounced skew) is like a catalog or inventory entry. For example, a generic ROLEX GOLD DAYTONA watch is a SKU, but the 3 specific ROLEX GOLD DAYTONA's listed on, for instance, EBAY™ are actual items or products.

As described above, in order to accommodate all of the possible products and services which might appear on a network and need to be listed and classified by the system, one needs to utilize a very large, complicated decision tree. Having to start at the top of that decision tree each time a product needs to be classified is inefficient. In order to reduce the amount of processing resources required, the present invention also provides mechanisms which allow a product to be partially classified based on information obtained from the site from which the product was obtained. Put another way, mechanisms serve to initially classify a product somewhere within the system where that initial classification is analyzed and the product is then further classified. This is achieved by defining a taxonomy for the site from which the product is obtained, and utilizing that taxonomy to partially classify the product within the system's taxonomy.

Sites such as YAHOO, EBAY, AMAZON, etc. arrange the products appearing in those sites in some logical order. This arrangement can be used as a scaffold for a taxonomy for those sites. By correlating categories of the taxonomy for a given site to the system's taxonomy, the identification of how a product is assigned within a given site's taxonomy can be used to assign that product somewhere within the system's taxonomy via the correlation. From

there, the decision nodes of the system's taxonomy can be used to further classify the product.

Referring back to Figure 4, the Taxonomy Module 420 is shown to include a primary table, called ***Taxonomy***. This table provides the description and organization of the system's taxonomy 425, as described above with regard to Figures 5 and 6. The ***Taxonomy*** table is substantially a static table once the system has come online. Periodically, the ***Taxonomy*** table is modified using a Taxonomy Administration Tool. Modification may be desired, for example, to add or modify the decision node queries; or add new decision nodes, branches and branch termini when it is identified that certain products are not being properly classified.

Optionally, two additional database tables may also be used for each product source site (e.g. YAHOO™, EBAY™, EXCITE™, etc.). In one embodiment, the two respective database tables may be stored on the same database server, such as on a Taxonomy server.

The first additional table is the ***xxx_Taxonomy***, where "xxx" is the site's name or designation as found in the ***Sites*** database table; for instance one example might be the "***YAHOO™_Taxonomy***".

As noted above, sites like YAHOO typically arrange products appearing in those sites in some logical order. This order native to the site is referred to as the "natural" taxonomy of that site. The ***xxx_Taxonomy*** table contains the "natural" taxonomy of the corresponding XXX site. In other words, the categorization found when navigating site XXX's auction or product listings directly is captured in site XXX's taxonomy table (***xxx_Taxonomy)***. This taxonomy table (***xxx_Taxonomy)*** may be built automatically by scanning the unique category names of ItemData for each site. Where necessary, the taxonomy table (***xxx_Taxonomy)*** may also be built manually or edited, as necessary.

The second additional table is called ***xxx_TaxonomyMap***, where "xxx" is the site's name or designation as found in the ***Sites*** database table; for instance one example might be the "***YAHOO™_TaxonomyMap***". This table serves to correlate a given ***xxx_Taxonomy*** table with the **Taxonomy** table. As

such, the ***xxx_TaxonomyMap*** operates as a mapping table, which contains correlations between a site's natural taxonomy (i.e., the site's natural "Taxonomy ID") and the system's taxonomy 425 (i.e., "Taxonomy ID"). This association mapping is generally inherently 'fuzzy' and may be thought of in terms of as a little 'hint' from the content editor.

The ***xxx_TaxonomyMap*** table is typically generated by a human "content editor" using a Taxonomy Admin Tool. However, different mechanisms (e.g., specialized software applications) may be implemented to automatically construct all or part of a given ***xxx_TaxonomyMap*** table.

A given site's taxonomy and taxonomy map tables tend to be fairly static. As described below, the classification module 415 may be configured to automatically update the site's taxonomy and taxonomy map tables if the classification module 415 encounters detects changes in the site's taxonomy, typically signaled by the identification of previously unseen categories.

In order to perform an initial classification, the classification module 415 uses category information supplied with an item or product (e.g., ItemData) to identify a taxonomy identification for the product for the site from which that product was obtained. In the event that the site's taxonomy map table does not have an entry for this category, a check is made to determine whether the site's **xxx_Taxonomy** table has this category. If this category did not exist in the site's taxonomy, it is added to the site's **xxx_Taxonomy** table (together with a flag indicating this category is new). The ItemData record for this item is left as unprocessed, and the next ItemData record is tested.

The corresponding **xxx_TaxonomyMap** table is then used to correlate the taxonomy identification from the product's site of origin with a taxonomy identification within the ***Taxonomy*** table. Provided that the site's taxonomy map has an entry for this category (i.e., the map has a unique category associated with this particular product's or item's category), the preliminary matching is accomplished, whereupon the process may apply a customized 'fuzzy' match technique logic to determine a more definitive match, i.e., identify the appropriate branch terminus or leaf for the product using the decision node queries.

In some instances, a category of a **xxx_Taxonomy** table may correlate to multiple categories of the system's taxonomy. This may occur, for example, if a site lists clothing, binoculars, and guns under a hunting category and these three categories are not in a category – subcategory relationship within the system's taxonomy. When there are multiple categories associated with a given site category, then each category which is correlated to that site category is considered in turn. For each category of the system's taxonomy, a set of keywords is selected such that if the site's taxonomy map table has keywords associated with this map entry, those keywords are used. Otherwise, the keywords from the category in the **xxx_TaxonomyMap** table may be used.

In one embodiment, a fuzzy premap operation is performed when a single site category maps to multiple categories of the system's taxonomy. In that case, the keywords used come from the xxx_TaxonomyMap table. As such, a first pass is made over all the potential "fuzzy postmap candidates", considering the text from the Title, and generally ignoring any guard words. If a match is found, then a recheck is made of the item using both Title and Description, as well as using the keywords. If the first title-only pass fails, the Description field will not be tested, and the classification module 415 will move on to the next fuzzy postmap candidate.

Once a single category of the system's taxonomy is selected as the "premap candidate", then a fuzzy postmap operation will activate, and it will use the keywords associated with the respective category (i.e., the queries associated with the decision nodes) in the system's taxonomy table 425 to find a single leaf category whose keywords match the incoming item Title and Description.

Typically, the system of the present invention only seeks to identify the first taxonomy category which matches the product. Once a match is identified, the remaining untested categories are ignored. If no category passes the Boolean query, then the item is either not classified or is assigned to a general category, and the process moves on to the next item in ItemData table 410.

As described previously with regard to Figures 5 and 6 above, the Boolean queries associated with the decision nodes leading from that

preliminary category are compared against the concatenation of the item's "Name" and "Description" fields. The process of using the Boolean queries associated with the decision nodes is also referred to herein as the 'fuzzy' match technique.

If the preliminary taxonomy category is a "leaf" (branch terminus) in the taxonomy table, then the preliminary category is unambiguous and becomes the final category under which this item or product is classified. The classified ItemData information is copied to the AllItems table 425, and the ItemData record is marked as having been processed.

If the preliminary taxonomy category is a tree level with multiple branches extending from the tree level, then the "fuzzy matching" technique or algorithm has to do a recursive walk of all the decision nodes extending from that tree level to find the best next level classification. The 'fuzzy matching', discussed in further detail below, generally operates as follows: (a) scan each decision node under this category (tree level); (b) using the queries associated with each decision node, perform a Boolean query against the concatenation of the item's NAME and DESCRIPTION. The first Boolean query of decision nodes which the product satisfies allows the product to be assigned to the next tree level related to that decision node.

The above process is continued for the next tree level until the Boolean query of the decision node which the product satisfies is a branch terminus or leaf. Once a leaf is reached, this becomes the final taxonomy category for this item, and processing will resume with the next ItemData record.

Each tree level typically includes a "General" decision node leading to a "General" leaf. If none of the other decision nodes on a tree level is satisfied, the "General" decision node is reached. The "General" decision node accepts all entries and thus allows the product to be assigned to the "General" leaf for the immediately preceding tree level. This becomes the item's (e.g., ItemData record's) final taxonomy category.

In one embodiment, the fuzzy mapping walk is recursive, meaning that if any tree level under a decision node that is satisfied is not a branch terminus and thus leads to multiple decision nodes, then the decision nodes extending

from the tree level are tested as well. In one embodiment, the recursive walk or algorithm is "depth first" so that it descends into increasingly lower tree levels once a given decision node is satisfied before testing other decision nodes on the same tree level, also referred to as peer categories. Depth first algorithms operate under the notion that if the content editor designed the system's taxonomy so that the best category match is one which resides the deepest within the system's taxonomy.

In one embodiment, once an item or product has been classified (e.g., classified ItemData) to some category in the system's taxonomy, it is marked as having been processed and a corresponding entry for the ItemData is made in the AllItems table 425.

In one embodiment, if an item or product (e.g., ItemData record) could not be assigned to a category, it is not marked as having been processed, and no entry is made for it in the AllItems table 425. In this way, the user does not view or otherwise see unclassified ItemData record. In an alternate embodiment, if an item or product (e.g., ItemData record) could not be assigned to a taxonomy category a "walk back" function is performed which moves up the table and looks for the first decision node which the product satisfied. This virtually guarantees each item will receive at least some General classification, whereupon the content editors may devise a classification arrangement to address why the item or product was sent to the General classification.

The Boolean queries used by the decision nodes may include a variety of different functionalities. For example, in one embodiment, the logic for the Boolean query comprises custom C++ code and does not involve an actual SQL query. While preferred Boolean query rules are set forth below, it is noted that these rules are only exemplary and that other rules may also be employed.

The keywords stored in the ***Taxonomy*** table and in each site's taxonomy map table are preferably configured to adhere to all, or any combination, of the following rules: (1) Keywords are separated by a space character (not tabs or commas); (2) Multiword phrases are enclosed in quotes (e.g., "like this"); (3) Keywords should be lower case (the query is not case sensitive, but the comparison routines assume lowercase for speed); (4) Any keyword may be

prefixed by a plus or a minus sign. This prefix is not separated from its keyword by any spaces; (5) Quoted keyword phrases should also have the "+" or "-" prefix, and appear immediately prior to the leading quote (not inside the quotes).

Preferred general rules for how these keywords are treated are as follows: (1) Any keyword prefixed by "+" should appear in the item's name or description for it to fit this category; (2) Any keyword prefixed by "-" should not appear in the item's name or description for it to fit this category; (3) If steps (1) and (2) are satisfied, the name or description field should contain at least one of any of the remaining (unprefixed) keywords. (4) Keyword order is generally not important, except for those quoted keyword phrases (which are treated as if they were a single word).

When matching a simple keyword string against words in the auction item's Name and Description, a very simple form of "stemming" is applied by allowing the keyword to be an incomplete match of the word. For example, the keyword "bead" would match the words "bead", "beads", "beaded", etc.. The keyword is allowed to be an incomplete match only on the trailing portion of the word. For example, the keyword "plane" would not match the word "airplane" because the keyword begins partway within the word.

Generally, at least one of any non-prefixed keyword must be present to match the name or description (e.g. "+a b c" indicates that "a" is required, and at least one of "b" or "c" must be present). However, one special exception is allowed; if the keyword expression contains no optional keywords (e.g. "+a –b"), then satisfying both the required ("+a") and forbidden ("-b") conditions is sufficient for a match. This means it is logically the same as if the user had written the expression as "+a –b a".

There are also special "wildcard" characters provided to allow for approximate keyword matches in the presence of what is exceedingly "noisy" data (e.g., the Name and Description for each auction item is entered by its respective seller, and the auction sites make no attempt to clean the data). Empirical studies showed that the spelling and punctuation provided by the sellers, while often wrong or inconsistent, at least followed some basic patterns,

and these patterns could be matched with some special forms of wildcard characters in the keywords. In one embodiment, these wildcards can be any or all of the following:

(1) The " – " (hyphen) character in a keyword is treated as a "soft separator". It will match a hyphen, a slash (forward or backward) or white space, or any combination. This allowed a keyword such as "15-in" to match "15-in", "15 in", "15-inch", "15 inch", "15/inch", etc..

(2) The " # " (pound) character will match any numeric character. This special character was introduced to allow approximate matching of numeric patterns such as model numbers, without requiring an explicit enumeration of all possible model numbers. For example, the keyword "##-in" would match any 2-digit number followed by an optional soft separator, and some form of "inch". The semantics of the " # " character require that the matching digits in the auction name or description do not appear as a substring within a larger numeric string. For example, "100-inch" would not match "##-in" because "100" contains too many numeric digits.

(3) The " @ " (at-sign) character will match only alphabetic characters. This character serves a similar role as " # " does for numbers, but it also allows more flexible alphabetic matching when multiple spellings are present. A good example is the string "POKEMON"™ which can be spelled with either a letter "e" or an "e-acute". Using a keyword such as `pok@mon` removes the need to have an exact match.

(4) The " ? " (question mark) character will match any single character of any type. In the POKEMON™ example above, an alternate acceptable keyword might be "pok?mon".

(5) The " | " (vertical bar) character will match any non-alphanumeric character (i.e. typically some punctuation or white space character). This capability becomes very important when it is necessary to

match an exact word rather than a partial substring. For example, whereas the keyword "bead" will match "bead", "beaded" or "beads", the keyword "bead" will match only "bead", as long as "bead" is the end of the string or is terminated by some non-alphanumeric character.

(6) The " * " (asterisk) wildcard has a special meaning – it will match anything (though this implementation does not permit " * " to be embedded, such as "pok*mon"). By convention, the " * " wildcard is used to indicate a "General" or "Miscellaneous" category to catch all auction items which did not satisfy any of the other categories' keywords.

One characteristic of items or products, such as auction items, is the fact that many can be uniquely identified by some make or model number. But rather than place a list of acceptable make and model numbers in each possible set of keywords, a "macro" facility may be introduced to allow for runtime "lookup" of such word lists. A macro may be identified by any string within a keyword pattern surrounded by the " % " (percent) character. For example, the keyword "%statenames%" identifies a macro by the name "statenames". When each auction's Name and Description are being evaluated for a match, the macro in the keyword is replaced by its respective replacement text. This text is found in a database table indexed by the macro name. In the example of "%statenames%", the database would contain a list of all 50 state names of the United States. Anywhere a state name needs to appear (or be blocked), a category's keywords can refer to the "%statenames%" macro. This not only reduces the amount of text associated with each category's keywords, but it facilitates maintenance, as a content editor needs to change the contents of the macro replacement string in one place to effectively change it for all categories.

In one embodiment, two special macros may be used to make it easier for a container category to include all the keywords from its child categories (without the content editor constantly maintaining this list). These special macros are "%children%" and "%allchildren%". The "%children%" macro

expands to include all the keywords from its immediate child categories, whereas "%allchildren%" recursively expands to include all the keywords from the child categories as well as the subcategories of those children (to any depth). When expanding the keywords for either of these two special macros, "guard words" (i.e.; keywords prefixed by " – ") are generally ignored, and "required words" (i.e.; keywords prefixed by " + ") are stripped of their " + " prefix. Essentially, all the childrens' keywords become optional works for this expansion.

In some instances, matches may only be made when the content editor has some form of "either-or" or "none of the above" functionality. In one embodiment, to meet this need, a special keyword syntax is allowed whereby keyword can be turned into separate expressions by enclosing them in parentheses. For example, a legal keyword might be "+(a b c) –(d e f)" when it is necessary to indicate that the name or description must contain at least one of "a" or "b" or "c", and it must not contain "d", "e" or "f".

Parenthetical keyword expressions are allowed to nest to arbitrary depths, and their order of evaluation follows the normal mathematical convention (inner expressions first). Each parenthetical expression is evaluated in a recursive fashion as if each were a separate keyword list. To leverage the power of the " + " (must be present) and " – " (cannot be present) operators allowed for normal keywords, these prefixes are also allowed in front of parenthetical expressions. The meaning of " + " and " – " in front of a parenthetical expression is enhanced to mean that the expression *must be true* (" + ") or it *must be false* (" – ").

When multiple parenthetical keyword expressions are present, their results are combined in a very predictable way:

1) If either expression contains a prefix (" + " or " – ") they are logically AND'ed.
2) If both of the expressions have no prefix, they are logically OR'ed.

Hence, for the keyword "(a b c) (d e f)", the 2 parenthetical expressions are OR'ed together, meaning the expression will be *true* if any of "a", "b", "c", "d", "e" or "f" is present (due to rule (2) above).

In the keyword expression "(a b c) –(d e f)", the expression will evaluate to *true* if and only if any of "a" or "b" or "c" is present AND as long as none of "d" or "e" or "f" are present. Similarly, the expression "+(a b c) –(d e f)" will evaluate to *true* if and only if at least one of "a" or "b" or "c" is present and none of "d", "e", or "f" is present.

Expressions in which parenthetical expressions are mixed with non-parenthetical expressions, all the non-parenthetical expressions are evaluated as if they were a single (unprefixed) parenthetical expression. In other words, the keyword expression "+a –b –(c d)" is logically equivalent to "(+a –b) –(c d)".

Some examples are as follow. Assuming a category has the keywords:

+hot -cold tea coffee "cup of"

then the following items will be categorized as describe below:

"milk"	Not classifiable. Does not contain "hot", or the optional "tea" or "coffee"
"hot chocolate"	Not classifiable. Matched "hot" but did not include "tea" or "coffee"
"tea"	Not classifiable. Doesn't contain required "hot" word
"hot or cold tea"	Not classifiable. Contained the forbidden "cold" keyword.
"hot tea"	Classifiable. Contains required "hot" and optional "tea".
"tea hot or warm"	Classifiable. Contains required "hot" and optional "tea" (order not important)
"hot"	Not classifiable. Contains required "hot", but no optional keywords
"hot cup of coffee"	Classifiable. Contains required "hot" and optional " cup of" and "coffee"

"hot coffee cup"	Classifiable. Contains required "hot" and optional "coffee"
"hot cups of coffee"	Classifiable. Contains required "hot" and optional "coffee"
"cups of hot water"	Not classifiable. Contains required "hot", but no optional keywords.
"cup of hot coffee"	Classifiable. Contains required "hot", optional "cup of" and optional "coffee"

As mentioned above, the system of the present invention applies what is called a "fuzzy" match technique when determining which category best describes product. The meaning of "fuzzy" in this context means that keywords can be used to assist in the classification process, and that there need not be a direct one-to-one map between, for example, an individual auction site's categories and the system's taxonomy categories.

As described above, there are potentially two kinds of fuzzy match implementations; a fuzzy "premap" and a fuzzy "postmap". The fuzzy "premap" is employed to correlate a product from a site's taxonomy table to the system's taxonomy table when there are more than one acceptable category within the system's taxonomy table for a given site category. In such instances, a content editor may have decided that products appearing in a given auction site's category belong to several different categories within the system's taxonomy table. As a result, the content editor correlates products from that category of the site's taxonomy table to multiple categories of the system's taxonomy table. When the classifier subsystem encounters this scenario, it evaluates the keywords for each candidate category within the system's taxonomy table against the Name and Description for that product. It does this using the standard process described above for evaluating keywords associated with each category.

It should be noted that the keywords associated with each category may be provided from a separate source and are not necessarily the same as the keywords associated with each category for normal processing. In other words, the content editor is free to choose a set of keywords to help the classifier 415 choose which of its several candidate categories is best. The classifier 415 typically chooses the first category whose keywords match.

Once a candidate category of the system's taxonomy is selected, either as the final result of the fuzzy "premap" or because the content editor mapped an auction site category directly to a single category, the fuzzy "postmap" process begins. The fuzzy postmap process is responsible for determining one and only one "leaf" category (also referred to herein as a branch terminus) under which a single item is to be classified. If the initial category which the postmap process evaluates is a leaf category (branch terminus), this process is completed. If the initial category is a reference category, i.e., the category refers the system to a different branch, the system uses as its new candidate the category to which the reference points. If the initial category has subcategories (meaning it is a tree level leading to multiple decision nodes), then this postmap process must evaluate the keywords associated with each decision node query for each subcategory to find the decision node which is satisfied.

It is noted with regard to reference categories that the process of following reference categories can be recursive. For example, a reference category can point to a tree level which includes a decision node leading to a reference category. The fuzzy postmap logic is responsible for following the satisfied decision nodes, no matter the path within the decision tree and regardless of the reference which it incorporates, until a leaf category is reached.

The fuzzy postmap logic is also responsible for detecting "loops" in this process (e.g. detecting a reference category which points to some container that ultimately points back at that the original reference).

"General" or "Miscellaneous" categories receive special treatment in this process. As noted previously, a general category is a branch terminus associated with a tree level via a decision node which accepts any products

which do not satisfy any of the other decision nodes associated with that tree level. If the classifier 415 encounters one of these general categories prior to evaluating all decision nodes associated with a tree level, the classifier 415 remembers the general category while it looks for a decision nodes associated with the tree level that is satisfied. Typically, only when the classifier has tested the product relative to all of the other decision nodes on a tree level will the classifier associate the product with the General category.

The outcome of this fuzzy postmap logic will generally be either a single leaf category under which a product can be classified or "no category found". In this latter case, the classifier 415 will consult the system's taxonomy category to which this site's category was mapped, and will look for the nearest "General" category. If none exists at that level in the system's taxonomy, the classifier goes up one tree level and sees if the product description satisfies any decision node associated with that product level. In this fashion, the classifier 415 walks back up the tree structure of the system's taxonomy table for the "nearest" general category associated with a tree level which the product satisfies. Provided the classifier 415 finds one, it becomes the branch terminus category under which the product is classified. If the product fails to become classified despite walking up the table, the item may be placed under a special "Unclassified" category.

The fuzzy postmap process may optionally consist of two separate passes over the set of candidate categories. According to this embodiment, during the first pass, only the item's Title (i.e. its brief description from the site) is analyzed for keyword matches. And in this pass, keywords prefixed by "-" are ignored. If a category satisfies this subset of keywords, then the item is re-evaluated using both the Title and the complete Description, and all keywords (including the " – " prefixed ones) are evaluated. Any category not satisfying this re-evaluation is removed or discarded as a candidate.

Provided the fuzzy postmap logic has considered all the subcategories within a single category without finding a valid match, a second pass over that set of subcategories will be invoked, this time evaluating all keywords against the combined Title and Description.

This two-pass technique enables the matching of items, such as auction items, by giving precedence to the Title. Sellers frequently list other items they are selling in the Description portion, such as the description portion of an auction item, which can lead to an ambiguous classification. By giving precedence to the Title (where the sellers generally do not mention other items), the classification process achieves a higher level of success in categorizing items. Within this fuzzy postmap portion associated with the classifier 415, "reference" categories play an important role. Basically, a "reference" category is one which merely points to another category. This provides a way for a content editor to detect that an auction item really belongs elsewhere in the system's taxonomy (e.g. it may have been mis-classified by the seller when the item was originally entered). The fuzzy postmap logic treats reference categories as a special form of "goto" when it is classifying each item. No auction item will ever be classified under a reference category, since reference categories are, by definition, not "leaf" or branch terminus categories.

Another feature of the system allows for any taxonomy category to be marked as "invisible on the web site". Categories marked in this way will not be presented to the web site browser. This property allows a content editor to hide categories which are not quite ready for production. It also allows the content editor to create "invisible reference" categories within the system's taxonomy 420. Such categories are just normal reference categories which are also marked "invisible". The classifier 415 treats these as it does any other reference category (by moving to the category to which it points to resume keyword evaluation).

In practice, these "invisible reference" categories generally serve as "forwarding addresses" which allow a content editor to detect that an auction item really needs to be classified elsewhere in the system's taxonomy 420, but does not want the web user to see this level of indirection. The fuzzy "premap" and "postmap" processes described above may be performed for all new items, such as auction items, found in the ItemData table 410 (as collected by the spider processes). This classification can be done at least once daily, or as often as necessary to keep the set of currently open auction items (e.g., classified

ItemData Records in the AllItems table 425) up to date. Classified auction items are moved to the AllItems table 425, which in one embodiment, is an archive containing all auction items, regardless of whether they are opened or closed. Periodically, however, it becomes necessary to "reclassify" archived auction items to maintain the most accurate pricing information. This reclassification may be implemented for any of the following reasons:

a) Content editors add, delete, or modify the system's taxonomy

b) Content editors add, delete, or modify the mappings between a site and the system's taxonomy

c) An auction site reorganizes its own "natural" taxonomy

d) Sellers modify their auction description or title after having submitted it, but before the item has sold.

Naturally, a complete reclassification of the archived items (e.g., classified ItemData records) can be a very costly process, because there could be many millions of items. To subset this problem and achieve better performance, an "incremental" form of reclassification may be provided.

The "incremental" reclassification determines a subset of items by detecting which of the system's taxonomy categories have been modified by the content editors. This may be accomplished by performing a correlated subquery against a transaction log of all changes made to either the system's taxonomy 420 or to any of the site-specific mapping tables. Every operation which alters the system's taxonomy 420 is recorded along with a timestamp. When it is time to reclassify the archived items, only those categories which have been modified since the last reclassification run need to be reconsidered.

The reclassification process may use the same fuzzy "premap" and "postmap" processes described above. The only significant difference is the source of the data. During normal classification, the data comes from ItemData (from the Spiders). During reclassification, the data come from AllItems (all the available or archived items or products).

E. Profiling Module 430

One of the powers of the system of the present invention is its ability to accumulate data, both present and historical, regarding the multitude of products which the system accumulates and classifies in its database. This enables the system to create a variety of different profiles about particular products and product categories which may be useful to buyers, sellers and manufacturers of merchandise. Because the data includes both present and historical data, the data can provide both current and historical perspectives. These perspectives may relate to the number of items sold or offered for sale. They may relate to price comparisons (average price, high price, low price, and price trends). The data can be used to identify when an item was first offered, as well as any increases or decreases in its sales as a function of time.

This type of data can be used by manufacturers to evaluate whether there is an attractive market for a product as well as provide pricing data for the product.

For sellers, the data can be used to evaluate how items should be priced and sold. For example, the data can be used to evaluate how items should be classified within a given site and how to best describe a given item. For instance, the accumulated data can be used to evaluate whether classifying a widget in one category on EBAY yields higher sales prices than classifying the same widget in a different category on EBAY. Describing an item with certain words may also be found to yield higher prices than other descriptions. One can also analyze the accumulated data to compare sales prices and sales success rates as a function of whether a same item is listed with or without a picture, or with or without a reserve. The data can also be used to evaluate whether there are preferred auction duration lengths as well as times that the auction should close.

For buyers, the data can be used to reveal advantageous bidding strategies. For example, the data can be used to suggest what is a recommended opening bid and what is a competitive closing bid.

The taxonomy system used in the present invention of classifying products adds further analytical capability to the data which the system accumulates. As noted previously, when an item is assigned to a particular branch terminus or leaf, the item is also inherently associated with the tree levels leading to that leaf. Each associated tree level can be viewed upon as a related category for the item. Hence, the same data analysis that may be conducted for a specific item (e.g., with regard to a specific leaf category), can also be done for all the items which are classified under the same tree level.

The profiling module 430, described herein, provides one more specific example of how the data accumulated by the system of the present invention can be leveraged.

The profiling module 430 may be configured to provide a valuation profile for a desired or selected item or product, based upon information accumulated for that corresponding identical item (e.g., accumulated classified ItemData). The same valuation profile may also be constructed for items which are associated with a particular tree level.

One application of the profiling module 430 is to analyze new identical products or items, as they are encountered, and update the system's taxonomy 420 to reflect the new pricing information. This pricing information may then be used by the system's taxonomy 420 to further facilitate classification of items. The profiling module 430 discharges this task by deciding, for each new identical item or product found, whether to modify the system's taxonomy 420 to reflect the change in the corresponding classified ItemData record or whether to leave it unchanged.

In one embodiment, the decision on whether the taxonomy 420 needs to be updated is based on an analysis of the Variance introduced to a particular leaf category if the item were assigned there. Variance is a statistical measure of the 'spread' in prices within that category. If the spread becomes too great, this evidences that the taxonomy 420 needs to be updated to create a new level of categories instead of lumping too many items together in one category. This issue is made difficult by the fact that the "normal" variance of each item may be different. An example from a different but related concept makes this clear.

The variance in prices of Internet stocks is much greater than public utility stocks, as is understood. Accordingly, the taxonomy 420 is also used to capture the acceptable variance by assigning a global variance, and then allowing the taxonomy 420 to relax that variance if necessary.

In one embodiment, by default, a global variance of 25% is applied to all branch termini. In other words, if prices of items filed under a particular branch terminus category are within 25% of each other, the items are deemed properly filed and no action is taken. If the variance is greater than 25%, the system either attempt to further subdivide the category into smaller groupings that have less variance. This subdivision may be done in an automated fashion or with the assistance of a system editor.

Alternatively, each branch terminus category within the taxonomy may be assigned its own unique variance threshold number. This may be accomplished by initially setting the variance threshold numbers to the same value. Alternately, the variance threshold numbers may be individually set as well.

As mentioned above, closed items are handled by requesting a GetClosed task or instruction from the spider registry 405B. The GetClosed task processing performed by a spider agent 405C checks to see whether an item or product actually sold, and if so, the price it sold at. Accordingly, the price at which an item or product sold is generally sent to the profiling module 430 and may be used by the profiling module 430 to construct a valuation profile for other products to which the sold product or item within that category.

Accordingly, provided a product associated with the ItemData record is no longer available (e.g., the product has been sold, offer period has lapsed, etc.), the profiling module 430 then processes the information from the spider agent 405C, referred to as post product information (PPI), which is information associated with a product that is no longer available (e.g., the product has been sold, offer period has lapsed, etc.). Accordingly, the profiling module 430 analyzes the PPI associated with the particular product or item that has been sold to determine valuation factors associated with the particular product. Correspondingly, in one embodiment, a closer module removes the ItemData

record from the AllItems table 425, if necessary, as the product is no longer available to a particular user for purchase, use, comparison, or consideration.

In one embodiment, the closer module functionality is provided by the spider module 405, wherein the spider module 405 is responsible for "noticing" which items (e.g., ItemData records) are no longer available and submitting a "close" record to remove the items (e.g., ItemData records) from the AllItems table 425. A stored procedure later sees that and cleans those items (e.g., ItemData records) out of the AllItems table 425.

In one embodiment, similar to the classification module 415, the profiling module 430 may classify the PPI according to the organization schema maintained within the system's taxonomy 420. As mentioned above, the taxonomy database comprises a series of categorized product descriptors which are broken down into a series of subcategory and micro-subcategory descriptors. As such, each product descriptor may be broken down into varying nth degrees of subcategories and micro-subcategories, and so on, as desired or required in order to accurately describe an individual product or item. Moreover, the selected search terms may include a variety of different search variables, including, but not limited to, price variables or prices ranges, age variables, condition variables (e.g., new or used), availability variables (e.g., 1 of 5 available), product offer expiration information, and a variety of other search variables as desired for product searching. As such, the classification or organization of the PPI may follow the organization shema (e.g., category, subcategory, and micro-subcategory) associated with the taxonomy database 425.

Accordingly, in one embodiment, the profiling module 430 collects the following valuation information associated with the each non-available product or item: (1) the market origination location of the product (e.g., URL location; e-auction site, electronic classified listings, etc.); (2) the price at which the product sold (e.g., sale price); the bid price range (e.g., high bid and low bid); (3) the condition of the product at sale (mint, average, poor); and (4) any additional other valuation information associated with the product, as desired. After compiling the valuation information of the corresponding non-available

product, the profiling module 430 may then perform different valuation operations on the valuation information of the associated non-available product.

In one embodiment, the profiling module 430 is configured to analyze the valuation information associated with each product to provide a valuation profile of the particular product under consideration based upon a valuation profile compiled from the corresponding identical item (e.g., ItemData records) as maintained by the system 400. Accordingly, the profiling module 430 averages the valuation of the individual product into the valuation records of previous, but identical, common items (e.g., ItemData records) information maintained by the system 400. As such, the profiling module 430 generates an average valuation profile record for each common item, wherein the average valuation profile for each common item comprises an average derived from the valuation information associated with non-available items or products (e.g., removed or expired ItemData records) corresponding to the particular common identical item.

In one embodiment, a min, max, average, and variance are computed and stored in a MetaItems table. Additionally, some statistical tools may be employed to recompute these numbers for currently archived items and to extract more interesting numbers (standard deviation, Z-scores, etc.), which may be used by the system 400.

Accordingly, the profiling module 430 may be configured to maintain an average valuation profile for each common item analyzed by the system 400. Accordingly, the profiling module 430 may be configured to present the average valuation profile for a particular common identical item (e.g., ItemData record) corresponding to a particular product or item in which the user may be interested in viewing. As such, the user can be presented with an average valuation corresponding to the same product in which the user is interested in, such that the user may compare a current product against the average valuation profile of the same corresponding or identical item (e.g., ItemData record) maintained by the system 400.

In one embodiment, the average valuation profile for an item (e.g., ItemData record), which would correspond to a particular selected identical

item or product, may include, but is not limited to, the following information: (1) the market origination location of the product (e.g., URL location; e-auction site, electronic classified listings, etc.); (2) a configurable time frame to be applied to the common product average valuation; (3) the average price at which past common products have sold; (4) the total number of common products sold in the past; (5) the bid price range (e.g., high bid and low bid); (6) the selling price of past common products (e.g., high price, low price, average retail price, average wholesale price, average used price, and average used price according to the condition of past common products); (7) product listing frequency; (8) seller bias (out of the total number sold, the ratio of merchant to seller sales: 32 Total= 22 merchant/10 individual); (9) regional bias (out of the total number sold, the location of the Top Seller Areas and the location of the Top Buyer Areas); (10) the number of searches conducted for this common product; (11) the system product search frequency (product searched for X number of times per time period – e.g., 2000 times / month); (12) the percentage of total system searches conducted for this product (e.g., ratio of product searched to all system searches – product searches/total system searches); (13) identification of the seller or offering part associated with the product or item; and / or a variety of other valuation information associated with and item or product, as desired. In one embodiment, the profiling module 430 provides a graphical representation of the average valuation profile for an item (e.g., ItemData record) versus the current valuation profile of a particular product or item being considered by a user.

As such, the profiling module 430 provides a user with tangible average valuation profile of an item (e.g., ItemData record) which corresponds to an identical product or item which is under consideration by the user, wherein the user is able to compare the valuation profile of a particular product or item against an average valuation profile, or product valuation history, of a corresponding identical item (e.g., ItemData record) maintained by the system 400.

As such, the system 400 provides an organized collection and categorization of products contained in a network, thereby providing a

comparative resource for comparing identical products or items that are available from a variety of different resources on a network. Additionally, in one embodiment, the system of the present invention is also configured to provide a valuation profile, which may be associated with a desired product or item selected by a user, wherein an interested user or consumer may compare the desired product or item price to a generated average valuation profile associated with the particular product or item. As such, the system 400 provides a comparative resource for comparing identical products or items to each other, as well as, an average valuation profile associated with each particular product or item.

4. Product Search Utilizing The System's Taxonomy

An objective of the system of the present invention is to aggregate vast amounts of information regarding products offered from a variety of sources on a network, such as the Internet, that are available for purchase, use, comparison, or consideration. A further objective of the system of the present invention is to enable users, most typically consumers, to search and analyze the product information that is aggregated.

A unique feature of the system of the present invention is its ability to organize vast amounts of disparate product data relative to a comprehensive hierarchical product classification system (i.e., the system's taxonomy) which finely categorizes the diverse range of products identified by the system and stored in its database. A further unique feature of the system of the present invention relates to how the system utilizes the same hierarchical product classification system to facilitate users to search and analyze the product information that the system aggregates. As a result, the system of the present invention is able to serve as an effective resource for comparing identical products or items that are available from a variety of different resources on a network.

As described above, the system's taxonomy is a well-organized, hierarchical product classification system consisting of numerous categories and

subcategories. In this section, it is discussed how the system supports searching and analyzing the collection of product information that the system aggregates using this product classification system to streamline the process of searching for products in the database. It is noted that the system may be accessed and searched via a network-based, preferably Internet-based service.

For a typical user of the Internet, one of the difficulties associated with Boolean searching is that one needs to know what one is looking for in advance. Many times, people search to identify an item where what they are looking for is not fully formulated in their heads. A further difficulty arises from the fact that people frequently do not know what things are called and/or do not know how to spell the words commonly used to describe items, such as brand names and model names. A further complexity arises from the fact that items can frequently be described in different ways, making it even harder for a person to effectively formulate a query to identify all the items he or she is looking for. Even if a person knows how to describe what one is looking for and knows all of the words to describe the item, the person typically does not have the skill or patience to formulate a complex query in order to effectively search for all of the items. The search methodology provided by the present invention circumvents many if not all of these difficulties.

As discussed with regard to Figures 5 and 6, each tree level 515A-D, 615A-D or branch terminus or leaf 530, 630 of the taxonomy serves as a product category. Tree levels represent product categories that may be further subdivided into narrower product categories, the narrower product categories being additional tree levels and/or branch termini of the taxonomy. If these narrower product categories are tree levels, they may be further subdivided into even narrower product categories, again being represented as additional tree levels and/or branch termini of the taxonomy.

Each tree levels and branch termini of the taxonomy has an associated decision node. The query associated with a given decision node serves to define the product category (or subcategory) that is intended to be represented by the tree level or branch terminus. As such, the query serves as a filter to exclude

products that do not fall within the scope of the category to be represented by the tree level or branch terminus.

In order for the query to serve as an effective filter, the query must be finely crafted with a great deal of content to differentiate items between different categories. As noted above, the queries attempt to capture all of the possible ways that someone might draft a product description to describe a particular product category. For example, in order to identify all product descriptions relating to watches, a query would include terms to cover the singular and plural forms of "watch", synonyms of "watch", misspellings of "watch" and its synonyms, brands of watches, and models of watches. As a result the queries can be >100 words long, optionally >500 words long and in some instances, >1000 words, >2000 words, >3000 words and even >4000 words in length.

A feature of the present invention is the ability of these decision node queries to also be used to facilitate product or product category searching. As such, the present invention takes advantage of the great level of detail incorporated into the decision node queries.

The system of the present invention is designed to accommodate millions of product descriptions, both current and past, in its database. These millions of product descriptions are classified into tens or hundreds of thousands of categories and subcategories with the assistance of the decision node queries. By Boolean searching decision node queries instead of product descriptions themselves, the system can reduce the number of text fields that need to be searched by at least an order of magnitude and potentially two orders of magnitude (depending on the ratio between the number of product descriptions in the database and the number of decision nodes employed in the taxonomy).

The taxonomy system is also used to assist the user to tailor their searches. Optionally, when an initial search is performed, the user is provided with a series of categories and subcategories corresponding to the categories of the system's taxonomy which match the user's query. Figure 7 illustrates a simplified taxonomy tree. Shown in bolded lines are the categories which a

hypothetical user's query satisfies. As can be seen, because the query is imprecise, the query can traverse many different unrelated branches.

By returning to the user as a search result the categories which match the user's query, the user is able to select one or more categories of the system's taxonomy which better match what the user was looking for. In this regard, the user is able to use the search of decision node queries and the matching categories to navigate the taxonomy to focus in on the items the user is seeking to identify. By reviewing the matching categories, the user is also able to identify keywords to help the user revise his or her query.

From the above description and drawings, it will be understood by those of ordinary skill in the art that the particular embodiments shown and described are for purposes of illustration only and are not intended to limit the scope of the invention. Those of ordinary skill in the art will recognize that the invention may be embodied in other specific forms without departing from its spirit or essential characteristics. References to details of particular embodiments are not intended to limit the scope of the claims.

1. A hierarchical product classification system comprising:

logic defining a branched decision tree comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; and

logic for classifying product entries having product descriptions with a branch terminus of the decision tree by assigning the product entry to a first tree level of the decision tree, and testing the product descriptions against decision node queries leading from the first tree level until a decision node query leading to a branch terminus is satisfied.

2. A hierarchical product classification system according to claim 1 wherein each tree level comprises at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

3. A hierarchical product classification system according to claim 1 wherein a portion of the branches direct the product description to a different branch when the decision node query associated with the branch is satisfied.

4. A hierarchical product classification system according to claim 1 wherein the decision tree comprises at least 100,000 branch termini.

5. A hierarchical product classification system according to claim 1 wherein the decision tree comprises at least 250,000 branch termini.

6. A hierarchical product classification system according to claim 1 wherein the decision tree comprises at least 500,000 branch termini.

7. A hierarchical product classification system according to claim 1 wherein the decision tree comprises at least 100,000 decision nodes.

8. A hierarchical product classification system according to claim 1 wherein the decision tree comprises at least 250,000 decision nodes.

9. A hierarchical product classification system according to claim 1 wherein the decision tree comprises at least 500,000 decision nodes.

10. A hierarchical product classification system according to claim 1 wherein the decision tree includes at least 10 tree levels extending between a first tree level and a branch terminus.

11. A hierarchical product classification system according to claim 1 wherein the decision tree includes at least 25 tree levels extending between a first tree level and a branch terminus.

12. A hierarchical product classification system according to claim 1 wherein the decision tree includes at least 50 tree levels extending between a first tree level and a branch terminus.

13. A hierarchical product classification system according to claim 1 wherein the decision tree includes at least 75 tree levels extending between a first tree level and a branch terminus.

14. A hierarchical product classification system according to claim 1 wherein the decision tree includes at least 100 tree levels extending between a first tree level and a branch terminus.

15. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries specify a plurality of brand and model names for a product category.

16. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries specify a negative limitation which if satisfied directs the classifying logic to exclude the product from the tree level or branch terminus extending from the decision node query.

17. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries are at least 100 words in length.

18. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries are at least 500 words in length.

19. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries are at least 1000 words in length.

20. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries are at least 2000 words in length.

21. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries are at least 3000 words in length.

22. A hierarchical product classification system according to claim 1 wherein at least a portion of the decision node queries are at least 4000 words in length.

23. An automated method for classifying a product entry within a hierarchical product classification system comprising:

taking a branched decision tree comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

wherein logic is associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or

branch terminus to which the branch associated with that decision node extends; and

classifying a product entry within the decision tree by

assigning the product entry to a first tree level of the decision tree,

employing the logic associated with decision node queries to test a product description associated with the product entry to be classified against the decision node queries leading from the first tree level until all of the decision node queries between the first tree level and a branch terminus are satisfied, and

classifying the product entry with the branch terminus whose decision node query was satisfied.

24. An automated method according to claim 23 wherein each tree level comprises at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

25. An automated method according to claim 23 wherein a portion of the branches direct the product description to a different branch when the decision node query associated with the branch is satisfied.

26. An automated method according to claim 23 wherein the decision tree comprises at least 100,000 branch termini.

27. An automated method according to claim 23 wherein the decision tree comprises at least 250,000 branch termini.

28. An automated method according to claim 23 wherein the decision tree comprises at least 500,000 branch termini.

29. An automated method according to claim 23 wherein the decision tree comprises at least 100,000 decision nodes.

30. An automated method according to claim 23 wherein the decision tree comprises at least 250,000 decision nodes.

31. An automated method according to claim 23 wherein the decision tree comprises at least 500,000 decision nodes.

32. An automated method according to claim 23 wherein the decision tree includes at least 10 tree levels extending between a first tree level and a branch terminus.

33. An automated method according to claim 23 wherein the decision tree includes at least 25 tree levels extending between a first tree level and a branch terminus.

34. An automated method according to claim 23 wherein the decision tree includes at least 50 tree levels extending between the first tree level and a branch terminus.

35. An automated method according to claim 23 wherein the decision tree includes at least 75 tree levels extending between the first tree level and a branch terminus.

36. An automated method according to claim 23 wherein the decision tree includes at least 100 tree levels extending between the first tree level and a branch terminus.

37. An automated method according to claim 23 wherein at least a portion of the decision node queries specify a plurality of brand and model names for a product category.

38. An automated method according to claim 23 wherein at least a portion of the decision node queries specify a negative limitation which if satisfied directs the classifying logic to exclude the product from the tree level or branch terminus extending from the decision node query.

39. An automated method according to claim 23 wherein at least a portion of the decision node queries are at least 100 words in length.

40. An automated method according to claim 23 wherein at least a portion of the decision node queries are at least 500 words in length.

41. An automated method according to claim 23 wherein at least a portion of the decision node queries are at least 1000 words in length.

42. An automated method according to claim 23 wherein at least a portion of the decision node queries are at least 2000 words in length.

43. An automated method according to claim 23 wherein at least a portion of the decision node queries are at least 3000 words in length.

44. An automated method according to claim 23 wherein at least a portion of the decision node queries are at least 4000 words in length.

45. An automated method according to claim 23 wherein the first tree level is manually selected from among the tree levels within the decision tree.

46. An automated method according to claim 23 wherein the first tree level is selected via computer executable logic from among the tree levels within the decision tree.

47. An automated method according to claim 23 wherein the first tree level is selected via computer executable logic from among the tree levels within the decision tree with the assistance of a mapping program which correlates the tree levels of the decision tree to a product organization of a location on a network from which the product entry is obtained.

48. An automated method according to claim 47 wherein the computer executable logic tests decision node queries associated with multiple different tree levels in order to select the first tree level.

49. An automated method according to claim 23 wherein the first tree level is randomly selected from among the tree levels within the decision tree.

50. An automated method according to claim 23, the method further comprising

identifying the product entry to be classified within the decision tree by spidering multiple different sites on a network for the product entry.

51. An automated method according to claim 23, wherein the automated method is repeated for over 500,000 different product entries.

52. An automated method according to claim 23, wherein the automated method is repeated for over 1,000,000 different product entries.

53. An automated method according to claim 23, wherein the automated method is repeated for multiple different product entries such that at least one branch terminus has at least 100 product entries classified with it.

54. An automated method according to claim 23, wherein the automated method is repeated for multiple different product entries such that at least one branch terminus has at least 500 product entries classified with it.

55. An automated method according to claim 23, wherein the automated method is repeated for multiple different product entries such that at least one branch terminus has at least 1,000 product entries classified with it.

56 An automated method according to claim 23, wherein the automated method is repeated for multiple different product entries such that at least one branch terminus has at least 10,000 product entries classified with it.

57 An automated method according to claim 23, wherein the automated method is repeated for multiple different product entries such that at least one branch terminus has at least 100,000 product entries classified with it.

58 A database comprising product information aggregated from multiple different sites on a network, the database comprising:

computer readable media encoding at least 100,000 different product entries identified from a variety of different sources on a network, each product entry having associated with it information identifying its assignment within a hierarchical product classification system comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

wherein a given product entry is classified with a branch terminus if a product description of the product entry satisfies the decision node query associated with the branch terminus.

59 A database according to claim 58 wherein no product entry is classified with more than one branch terminus.

60. A database according to claim 58 wherein the database comprises at least 1,000,000 different product entries.

61. A database according to claim 60 wherein no product entry is classified with more than one branch terminus.

62. A database according to claim 58 wherein at least one product entry in the database is associated with at least 100,000 different branch termini.

63. A database according to claim 58 wherein at least one product entry in the database is associated with at least 250,000 different branch termini.

64. A database according to claim 58 wherein at least one product entry in the database is associated with at least 500,000 different branch termini.

65. A database according to claim 58 wherein at least 1,000 product entries have been assigned to at least one branch terminus.

66. A database according to claim 58 wherein at least 10,000 product entries have been assigned to at least one branch terminus.

67. A database according to claim 58 wherein at least 100,000 product entries have been assigned to at least one branch terminus.

68. A product search system comprising:

logic defining a branched decision tree comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends;

logic for classifying product entries having product descriptions with a branch terminus of the decision tree by assigning the product entry to a first tree level of the decision tree, and testing the product descriptions against decision node queries leading from the first tree level until a decision node query leading to a branch terminus is satisfied; and

logic for searching the classified product entries by

matching the decision node queries to a search query,

identifying tree level categories and branch terminus categories having decision node queries which match the search query, and

returning product categories relating to the identified tree level categories and branch terminus categories.

69. A product search system according to claim 68, further including returning product entries belonging to the identified tree level categories and branch terminus categories.

70. A product search system according to claim 68, further including logic for returning product entries relating to one or more product categories selected from among the returned product categories.

71. A product search system according to claim 68 wherein each tree level comprises at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

72. A product search system according to claim 68 wherein a portion of the branches direct the product description to a different branch when the decision node query associated with the branch is satisfied.

73. A product search system according to claim 68 wherein the decision tree comprises at least 100,000 branch termini.

74. A product search system according to claim 68 wherein the decision tree comprises at least 250,000 branch termini.

75. A product search system according to claim 68 wherein the decision tree comprises at least 500,000 branch termini.

76. A product search system according to claim 68 wherein the decision tree comprises at least 100,000 decision nodes.

77. A product search system according to claim 68 wherein the decision tree comprises at least 250,000 decision nodes.

78. A product search system according to claim 68 wherein the decision tree comprises at least 500,000 decision nodes.

79. A product search system according to claim 68 wherein the decision tree includes at least 10 tree levels extending between a first tree level and a branch terminus.

80. A product search system according to claim 68 wherein the decision tree includes at least 25 tree levels extending between a first tree level and a branch terminus.

81. A product search system according to claim 68 wherein the decision tree includes at least 50 tree levels extending between a first tree level and a branch terminus.

82. A product search system according to claim 68 wherein the decision tree includes at least 75 tree levels extending between a first tree level and a branch terminus.

83. A product search system according to claim 68 wherein the decision tree includes at least 100 tree levels extending between a first tree level and a branch terminus.

84. A product search system according to claim 68 wherein at least a portion of the decision node queries specify a plurality of brand and model names for a product category.

85. A product search system according to claim 68 wherein at least a portion of the decision node queries specify a negative limitation which if satisfied directs the classifying logic to exclude the product from the tree level or branch terminus extending from the decision node query.

86. A product search system according to claim 68 wherein at least a portion of the decision node queries are at least 100 words in length.

87. A product search system according to claim 68 wherein at least a portion of the decision node queries are at least 500 words in length.

88. A product search system according to claim 68 wherein at least a portion of the decision node queries are at least 1000 words in length.

89. A product search system according to claim 681 wherein at least a portion of the decision node queries are at least 2000 words in length.

90. A product search system according to claim 68 wherein at least a portion of the decision node queries are at least 3000 words in length.

91. A product search system according to claim 68 wherein at least a portion of the decision node queries are at least 4000 words in length.

92. An automated method for classifying a product entry within a hierarchical product classification system comprising:

taking a branched decision tree comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

wherein logic is associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends;

classifying a product entry within the decision tree by

assigning the product entry to a first tree level of the decision tree,

employing the logic associated with decision node queries to test a product description associated with the product entry to be classified against the decision node queries leading from the first tree level until all of the decision node queries between the first tree level and a branch terminus are satisfied, and

classifying the product entry with the branch terminus whose decision node query was satisfied; and

searching the classified product entries by having computer executable logic

match the decision node queries to a search query,

identify tree level categories and branch terminus categories having decision node queries which match the search query, and

return product categories relating to the identified tree level categories and branch terminus categories.

93. A product search method according to claim 92, the method further including displaying product entries belonging to the identified tree level categories and branch terminus categories.

94. A product search method according to claim 92 wherein each tree level comprises at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading

to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

95. A product search method according to claim 92 wherein a portion of the branches direct the product description to a different branch when the decision node query associated with the branch is satisfied.

96. A product search method according to claim 92 wherein the decision tree comprises at least 100,000 branch termini.

97. A product search method according to claim 92 wherein the decision tree comprises at least 250,000 branch termini.

98. A product search method according to claim 92 wherein the decision tree comprises at least 500,000 branch termini.

99. A product search method according to claim 92 wherein the decision tree comprises at least 100,000 decision nodes.

100. A product search method according to claim 92 wherein the decision tree comprises at least 250,000 decision nodes.

101. A product search method according to claim 92 wherein the decision tree comprises at least 500,000 decision nodes.

102. A product search method according to claim 92 wherein the decision tree includes at least 10 tree levels extending between a first tree level and a branch terminus.

103. A product search method according to claim 92 wherein the decision tree includes at least 25 tree levels extending between a first tree level and a branch terminus.

104. A product search method according to claim 92 wherein the decision tree includes at least 50 tree levels extending between the first tree level and a branch terminus.

105. A product search method according to claim 92 wherein the decision tree includes at least 75 tree levels extending between the first tree level and a branch terminus.

106. A product search method according to claim 92 wherein the decision tree includes at least 100 tree levels extending between the first tree level and a branch terminus.

107. A product search method according to claim 92 wherein at least a portion of the decision node queries specify a plurality of brand and model names for a product category.

108. A product search method according to claim 92 wherein at least a portion of the decision node queries specify a negative limitation which if satisfied directs the classifying logic to exclude the product from the tree level or branch terminus extending from the decision node query.

109. A product search method according to claim 92 wherein at least a portion of the decision node queries are at least 100 words in length.

110. A product search method according to claim 92 wherein at least a portion of the decision node queries are at least 500 words in length.

111. A product search method according to claim 92 wherein at least a portion of the decision node queries are at least 1000 words in length.

112. A product search method according to claim 92 wherein at least a portion of the decision node queries are at least 2000 words in length.

113. A product search method according to claim 92 wherein at least a portion of the decision node queries are at least 3000 words in length.

114. A product search method according to claim 92 wherein at least a portion of the decision node queries are at least 4000 words in length.

115. A product search method according to claim 92 wherein the first tree level is manually selected from among the tree levels within the decision tree.

116. A product search method according to claim 92 wherein the first tree level is selected via computer executable logic from among the tree levels within the decision tree.

117. A product search method according to claim 92 wherein the first tree level is selected via computer executable logic from among the tree levels within the decision tree with the assistance of a mapping program which correlates the tree levels of the decision tree to a product organization of a location on a network from which the product entry is obtained.

118. A product search method according to claim 117 wherein the computer executable logic tests decision node queries associated with multiple different tree levels in order to select the first tree level.

119. A product search method according to claim 92 wherein the first tree level is randomly selected from among the tree levels within the decision tree.

120. A system for statistically profiling product information comprising:

logic defining a branched decision tree comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

logic associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends;

logic for classifying product entries having product descriptions with branch termini of the decision tree by assigning the product entries to initial tree levels of the decision tree, and testing the product descriptions against decision node queries leading from the initial tree levels until a decision node query leading to a branch terminus is satisfied; and

logic for statistically profiling a group of classified product entries that relate to a selected product subcategory by

taking a tree level associated with that product subcategory, identifying product entries classified with branch termini which extend from that tree level, and

performing a statistical analysis of at least one property associated with the identified product entries.

121. A system according to claim 120 wherein the at least one property is selected from the group consisting of a number of same or similar product entries currently available; a number of same or similar product entries sold; a highest price for product entries sold in that subcategory; a lowest price for product entries sold in that subcategory; an average price for product entries sold in that subcategory; a median price for product entries sold in that subcategory; a first date a product entry in that subcategory was listed on the network; a first date a product entry in that subcategory was sold on the network; a most recent date a product entry in that subcategory was listed on the network; a most recent date a product entry in that subcategory was sold on the network; a frequency that product entries in that subcategory are listed on the network; a frequency that product entries in that subcategory are sold on the network; and sales price trend information for product entries in that subcategory.

122. A system according to claim 120 wherein each tree level comprises at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

123. A system according to claim 120 wherein a portion of the branches direct the product description to a different branch when the decision node query associated with the branch is satisfied.

124. A system according to claim 120 wherein the decision tree comprises at least 100,000 branch termini.

125. A system according to claim 120 wherein the decision tree comprises at least 250,000 branch termini.

126. A system according to claim 120 wherein the decision tree comprises at least 500,000 branch termini.

127. A system according to claim 120 wherein the decision tree comprises at least 100,000 decision nodes.

128. A system according to claim 120 wherein the decision tree comprises at least 250,000 decision nodes.

129. A system according to claim 120 wherein the decision tree comprises at least 500,000 decision nodes.

130. A system according to claim 120 wherein at least a portion of the decision node queries are at least 100 words in length.

131. A system according to claim 120 wherein at least a portion of the decision node queries are at least 500 words in length.

132. A system according to claim 120 wherein at least a portion of the decision node queries are at least 1000 words in length.

133. A system according to claim 120 wherein at least a portion of the decision node queries are at least 2000 words in length.

134. A system according to claim 120 wherein at least a portion of the decision node queries are at least 3000 words in length.

135. A system according to claim 120 wherein at least a portion of the decision node queries are at least 4000 words in length.

136. An automated method for statistically profiling product information comprising:

taking a branched decision tree comprising

a plurality of main branches corresponding to different categories of products,

a plurality of tree levels extending from each main branch, each tree level corresponding to a subcategory of the category of products to which a related main branch corresponds,

a plurality of branches extending from each tree level of the decision tree, each branch leading to either another tree level or a branch terminus, the branch terminus corresponding to a product category on the decision tree to which a product entry to be classified may be assigned, and

a plurality of decision nodes, each decision node interconnecting a branch leading to either another tree level or a branch terminus to the tree level from which the branch extends,

wherein logic is associated with each decision node defining a query that needs to be satisfied by a product description of a product entry to be classified in order for that product entry to be classified with the tree level or branch terminus to which the branch associated with that decision node extends; and

classifying product entries within the decision tree by

assigning the product entries to a first tree level of the decision tree,

employing the logic associated with decision node queries to test a product description associated with the product entries to be classified against the decision node queries leading from the first tree level until all of the decision node queries between the first tree level and a branch terminus are satisfied, and

classifying the product entries with the branch terminus whose decision node query was satisfied.

statistically profiling a group of classified product entries that relate to a selected product subcategory by

taking a tree level associated with that product subcategory, identifying product entries classified with branch termini which extend from that tree level, and

performing a statistical analysis of at least one property associated with the identified product entries.

137. An automated method according to claim 136 wherein the at least one property is selected from the group consisting of a number of same or similar product entries currently available; a number of same or similar product entries sold; a highest price for product entries sold in that subcategory; a lowest price for product entries sold in that subcategory; an average price for product entries sold in that subcategory; a median price for product entries sold in that subcategory; a first date a product entry in that subcategory was listed on the network; a first date a product entry in that subcategory was sold on the network; a most recent date a product entry in that subcategory was listed on the network; a most recent date a product entry in that subcategory was sold on the network; a frequency that product entries in that subcategory are listed on the network; a frequency that product entries in that subcategory are sold on the network; and sales price trend information for product entries in that subcategory.

138. An automated method according to claim 136 wherein each tree level comprises at least one branch extending from the tree level leading to a branch terminus, the decision node query associated with the branch leading to the branch terminus being satisfied by the product description not satisfying any of the other decision node queries extending from the tree level.

139. An automated method according to claim 136 wherein a portion of the branches direct the product description to a different branch when the decision node query associated with the branch is satisfied.

140. An automated method according to claim 136 wherein the decision tree comprises at least 100,000 branch termini.

141. An automated method according to claim 136 wherein the decision tree comprises at least 250,000 branch termini.

142. An automated method according to claim 136 wherein the decision tree comprises at least 500,000 branch termini.

143. An automated method according to claim 136 wherein the decision tree comprises at least 100,000 decision nodes.

144. An automated method according to claim 136 wherein the decision tree comprises at least 250,000 decision nodes.

145. An automated method according to claim 136 wherein the decision tree comprises at least 500,000 decision nodes.

146. An automated method according to claim 136 wherein at least a portion of the decision node queries are at least 100 words in length.

147. An automated method according to claim 136 wherein at least a portion of the decision node queries are at least 500 words in length.

148. An automated method according to claim 136 wherein at least a portion of the decision node queries are at least 1000 words in length.

149. An automated method according to claim 136 wherein at least a portion of the decision node queries are at least 2000 words in length.

150. An automated method according to claim 136 wherein at least a portion of the decision node queries are at least 3000 words in length.

151. An automated method according to claim 136 wherein at least a portion of the decision node queries are at least 4000 words in length.

152. An automated method according to claim 136 wherein classifying product entries includes classifying at least 500,000 different product entries.

153. An automated method according to claim 136 wherein classifying product entries includes classifying at least 1,000,000 different product entries.

154. An automated method according to claim 136 wherein at least one branch terminus has at least 100 product entries classified with it.

155. An automated method according to claim 136 wherein at least one branch terminus has at least 1,000 product entries classified with it.

156. An automated method according to claim 136 wherein at least one branch terminus has at least 10,000 product entries classified with it.

157. An automated method according to claim 136 wherein at least one branch terminus has at least 100,000 product entries classified with it.

158. An automated method according to claim 136 wherein the group of classified product entries profiled includes at least 100 different product entries.

159. An automated method according to claim 136 wherein the group of classified product entries profiled includes at least 1,000 different product entries.

160. An automated method according to claim 136 wherein the group of classified product entries profiled includes at least 10,000 different product entries.

161. An automated method according to claim 136 wherein the first tree level is selected via computer executable logic from among the tree levels within the decision tree.

162. An automated method according to claim 136 wherein the first tree level is selected via computer executable logic from among the tree levels within the decision tree with the assistance of a mapping program which correlates the tree levels of the decision tree to a product organization of a location on a network from which the product entry is obtained.

163. An automated method according to claim 162 wherein the computer executable logic tests decision node queries associated with multiple different tree levels in order to select the first tree level.

164. An automated method according to claim 136 wherein the first tree level is randomly selected from among the tree levels within the decision tree.

165. An automated method according to claim 136, the method further comprising

identifying the product entries to be classified within the decision tree by spidering multiple different sites on a network for the product entry.

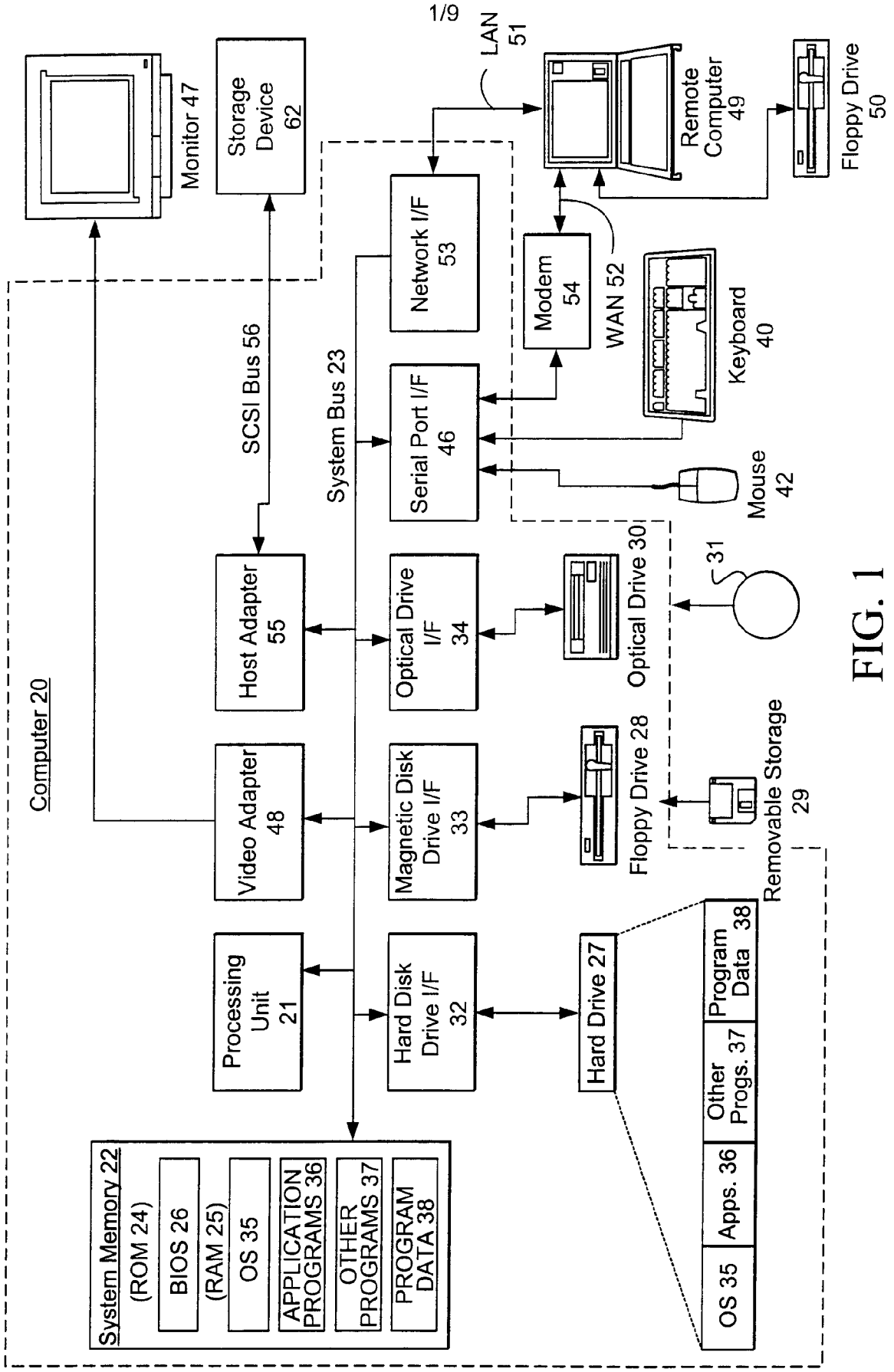


FIG. 1

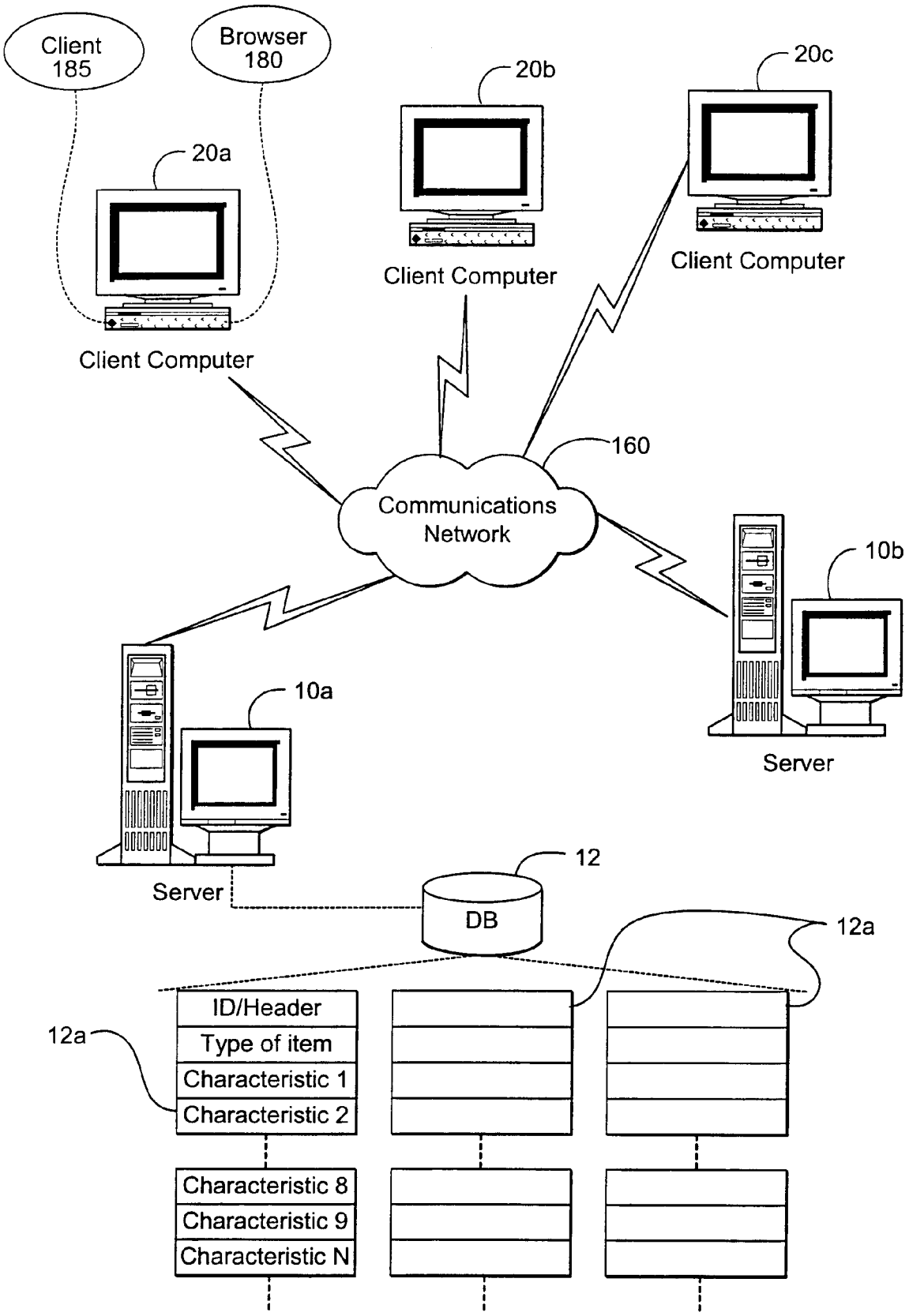
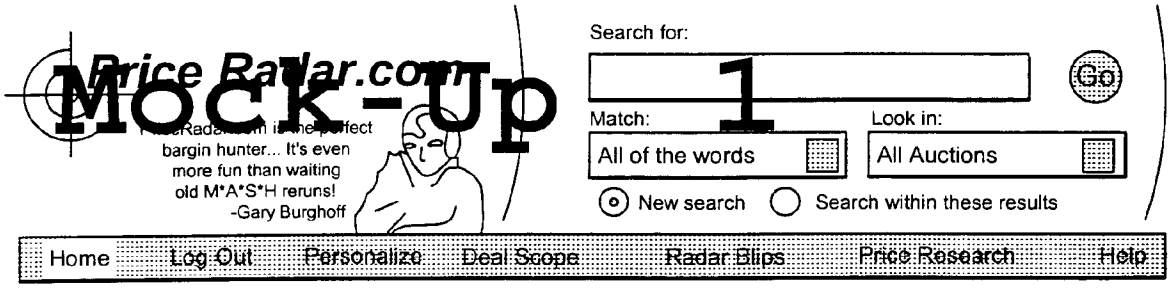

Client 185
Browser 180
20a
20b
20c
Client Computer
Client Computer
Client Computer
160
Communications Network
10a
10b
Server
Server
12
DB
12a
12a
ID/Header
Type of item
Characteristic 1
Characteristic 2
Characteristic 8
Characteristic 9
Characteristic N

FIG. 2



Welcome Steve! Your tracking 7 items on your Deal Scope, and 4 items in Radar Blips

Browse by Category

- Antiques (61,052).
- Beanie Babies (75,269).
- Cars, Trucks, etc (17,541).
- Coins & Stamps (97,820).
- Collectibles (476,645).
- Computers (114,276).
- Dolls & Figures (92,274).
- Electronics (48,581).
- Entertainment (407,674.
- Glass, Porcelain & Pottery (159,835).
- Home & Leisure (154,494).
- Jewelry (118,004).
- Memorabilia (243,728).
- Miscellaneous (17,373).
- Sports (35.852).
- Sports Memorabilia (343,773).
- Toys (151,060).
- Travel (1,192).

Featured Auctions

- Domain Names
- Action Figures
- Pokemon

About Our Services

PriceRadar.com is a free service that gives you the lorem ipsum dolor sit am et, consectetuer adip elit, sed diam nonummy nibh edi smod tincidunt ut laoreet: dolore magna aliquam er at: volutpat: Ut: wisi enim ad minim veniam quis nostrud exrci tation ullam corper susipit labortis nist ut aliquip ex ea comm odo consequat.

New to PriceRadar
Click here to
Get Started

Deal Scope™
Tracks all your on-line auctions and items of interest from one master control center

Radar Blips™
Gets daily e-mail notifications when items of interest appear on any auction site

In hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla faclisis at vero eros et accum san et iusto odio dignissim qui blandit praesent luptatum

What's New

- **Aquilam erat**

Lorem ipsum dolor sit amet, consect etuer adipiscomg elit, sed diam nonum my nibh euismod tincidunt ut

- **PriceRadar Newsletter**

Hot auction tips and strategies, ntial ion of new categories and services, plus much much more

Sign up for the
Weekly Dish

PriceRadar's weekly newsletter and stay informed

your e-mail address

plain text HTML

privacy policy

FIG. 3A

FIG. 3B

Price Radar.com

Search for: Go

Match: All of the words | Look in: ebay

Browser: All Categories

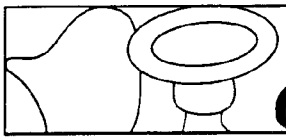
New search | Search within these results

Home | Log Out | Personalize | Radar Scope | Radar Blips | Price Research | Help

You are in: Price Search Analysis

Mock-Up 2

Analysis | Professional Appraisal | BHI Calculator



Glass

You just might find it at ebay™

Selected Item: Rolex Explorer 2

On Auction @ ebay Item # : 147053150 Current Price $2,225

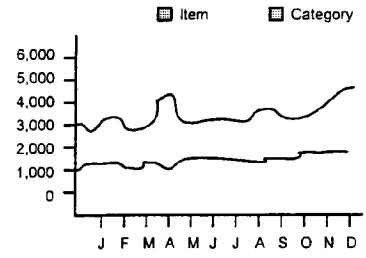
Parent Category: Jewelry > Watches > High End

Price History

Look at: Auctions | Time Period: 1 Year

	Selected Item	Parent Category
Average Price	$2,674	$1,388
Total Number Sold	32	198
High Price	$3,425	$5,290
Low Price	$1,900	$1,001
COMPARISONS:		
Avg. Retail Price	$3,900	$3,900
Avg. Wholesale Price	$1,840	$1,840
Avg. Used Price		
Mint Condition	$2,952	$2,952
Avg. Condition	$2,230	$2,230
Poor Condition	$1,867	$1,867

Average Price Over Time :



Item Statistics

Listing Frequency :	2.66/mo.
Total Number Sold :	32
Seller Bias (merchant/person) :	22/10
Regional Bias :	
Top 3 Seller Areas :	
1. Los Angles, CA.	
2. Chicago, Ill.	
3. Petaluma, CA.	
Top 3 Buyers Areas :	
1. New York, NY.	
2. Fort Lauderdale, FL.	
3. San Francisco, CA.	
PriceRadar Searches for this Item :	14.786
PriceRadar Search Frequency :	1,232/mo.
Percent of All PriceRadar Searches :	.04%

Search the Web for More Information

Need consumer information, a look at the manufacturer's Web site, distributor pricing and availability? Find it all here...

Search for: Rolex Explorer 2

Search Site: Alta Vista | Go

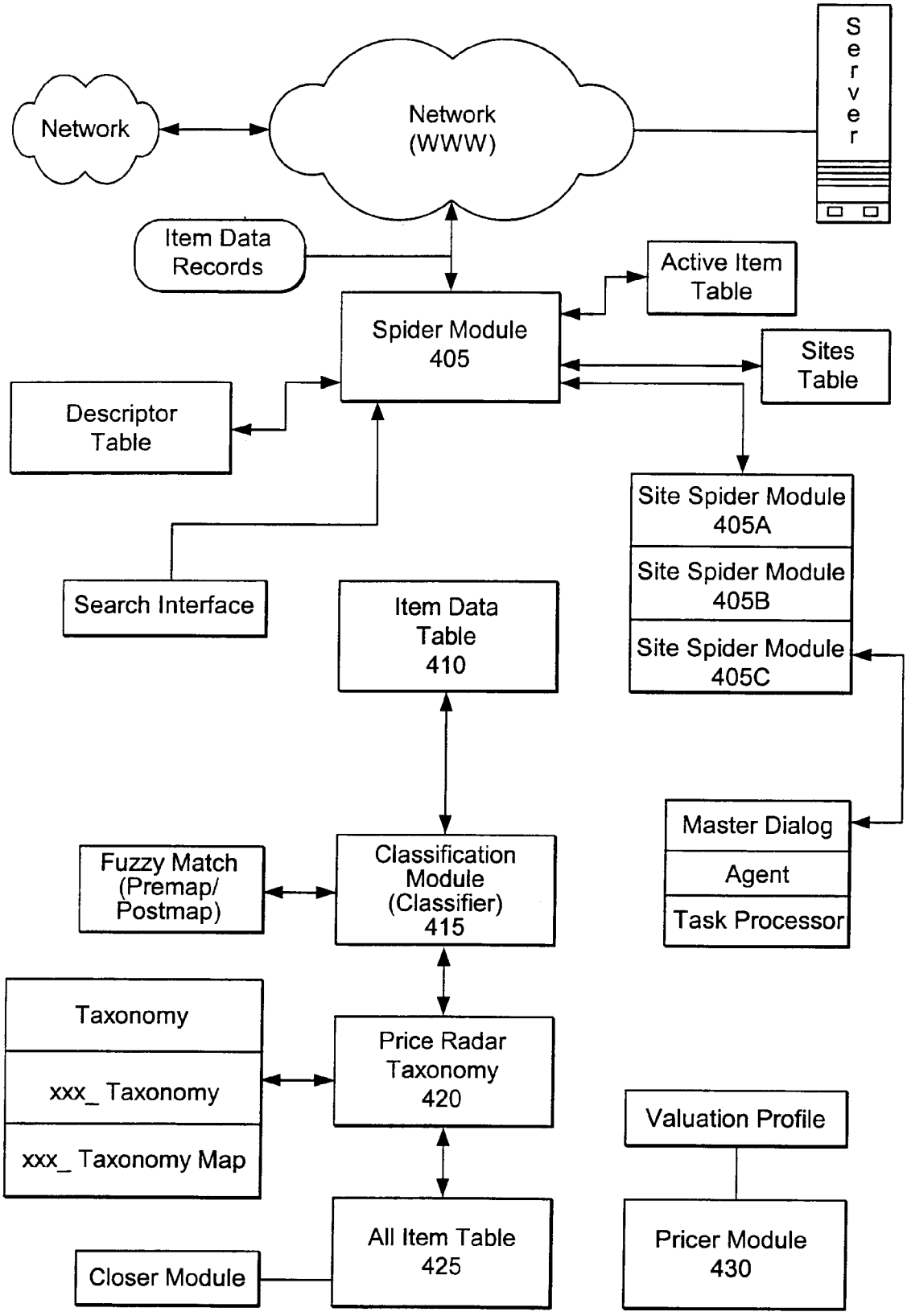

Network
Network (WWW)
Server
Item Data Records
Active Item Table
Spider Module 405
Sites Table
Descriptor Table
Site Spider Module 405A
Site Spider Module 405B
Site Spider Module 405C
Search Interface
Item Data Table 410
Master Dialog
Agent
Task Processor
Fuzzy Match (Premap/ Postmap)
Classification Module (Classifier) 415
Taxonomy
xxx_ Taxonomy
xxx_ Taxonomy Map
Price Radar Taxonomy 420
Valuation Profile
All Item Table 425
Closer Module
Pricer Module 430

FIG. 4A

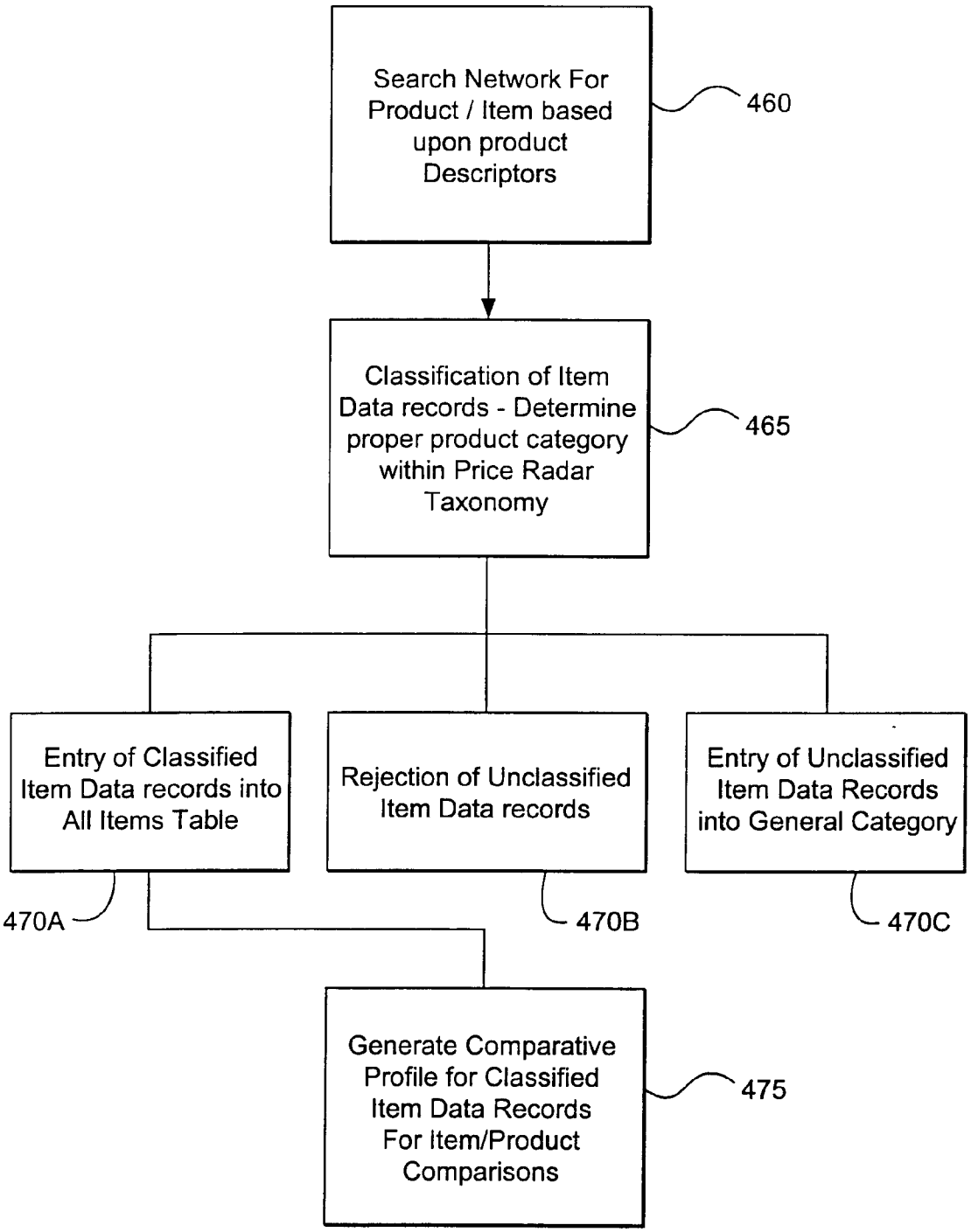

Search Network For Product / Item based upon product Descriptors
460
Classification of Item Data records - Determine proper product category within Price Radar Taxonomy
465
Entry of Classified Item Data records into All Items Table
Rejection of Unclassified Item Data records
Entry of Unclassified Item Data Records into General Category
470A
470B
470C
Generate Comparative Profile for Classified Item Data Records For Item/Product Comparisons
475

FIG. 4B

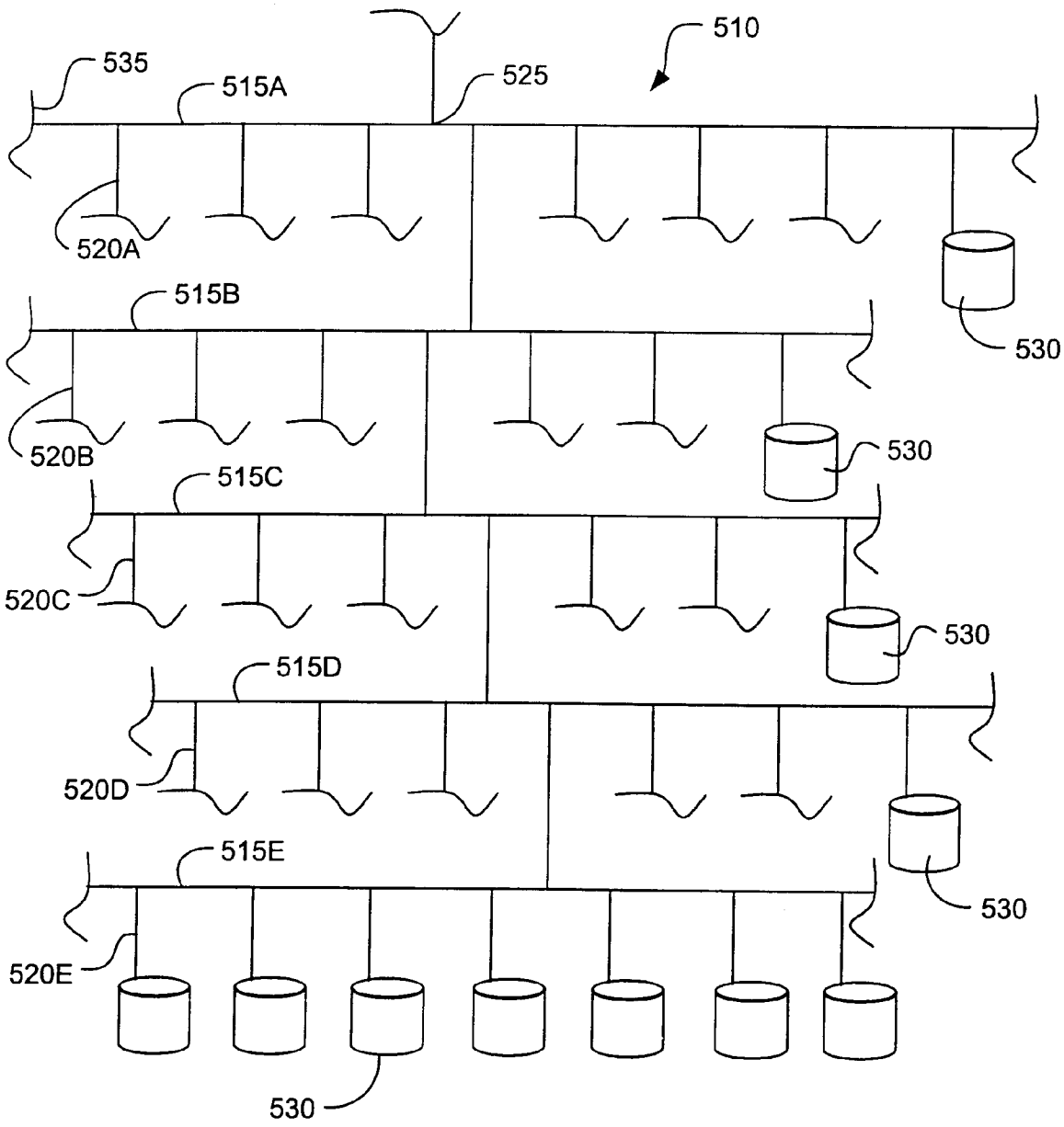

510
535
515A
525
520A
515B
530
520B
530
515C
520C
530
515D
520D
515E
530
520E
530

FIG. 5

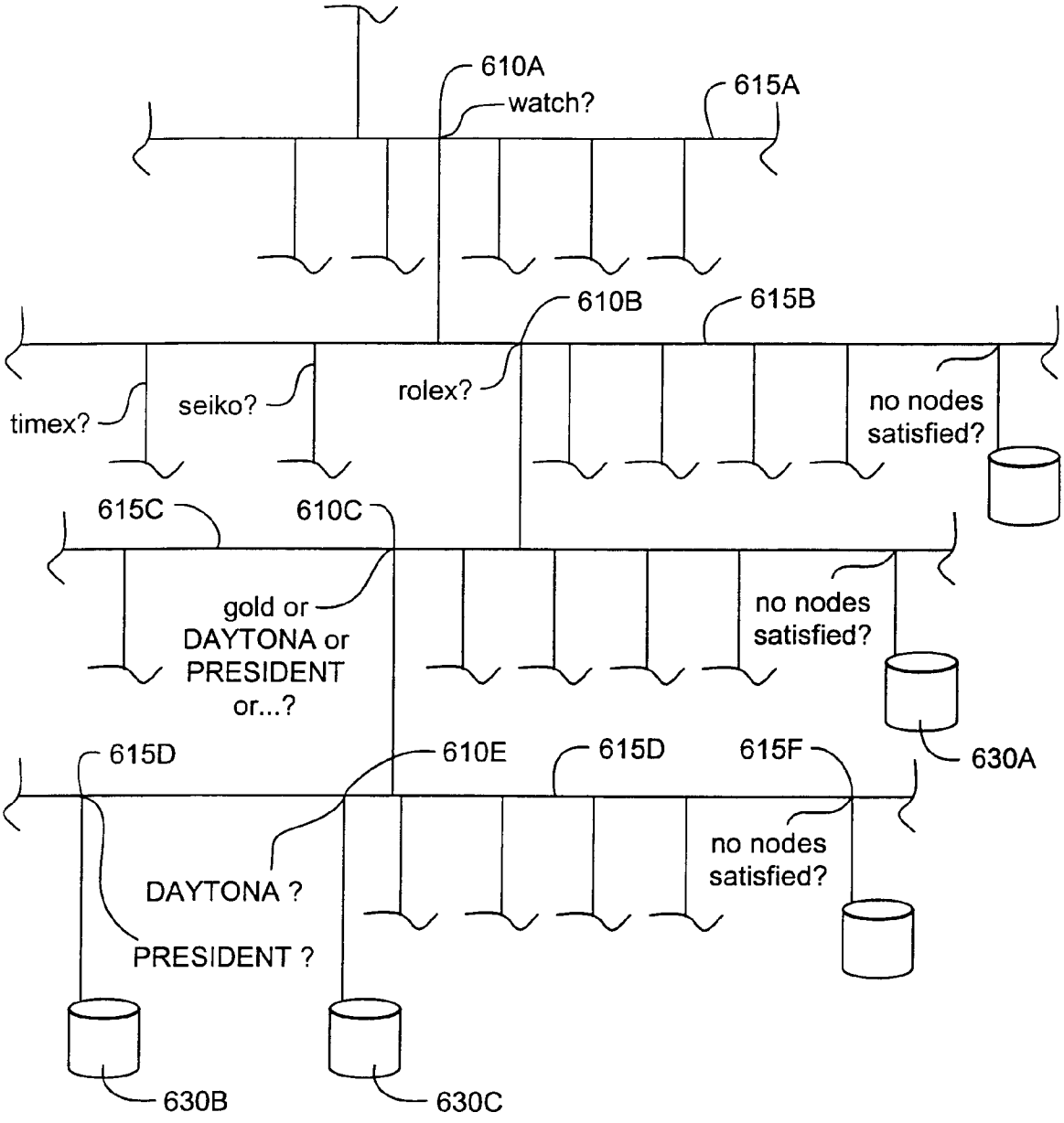

610A
watch?
615A
610B
615B
timex?
seiko?
rolex?
no nodes satisfied?
615C
610C
gold or DAYTONA or PRESIDENT or...?
no nodes satisfied?
630A
615D
610E
615D
615F
DAYTONA ?
PRESIDENT ?
no nodes satisfied?
630B
630C

FIG. 6

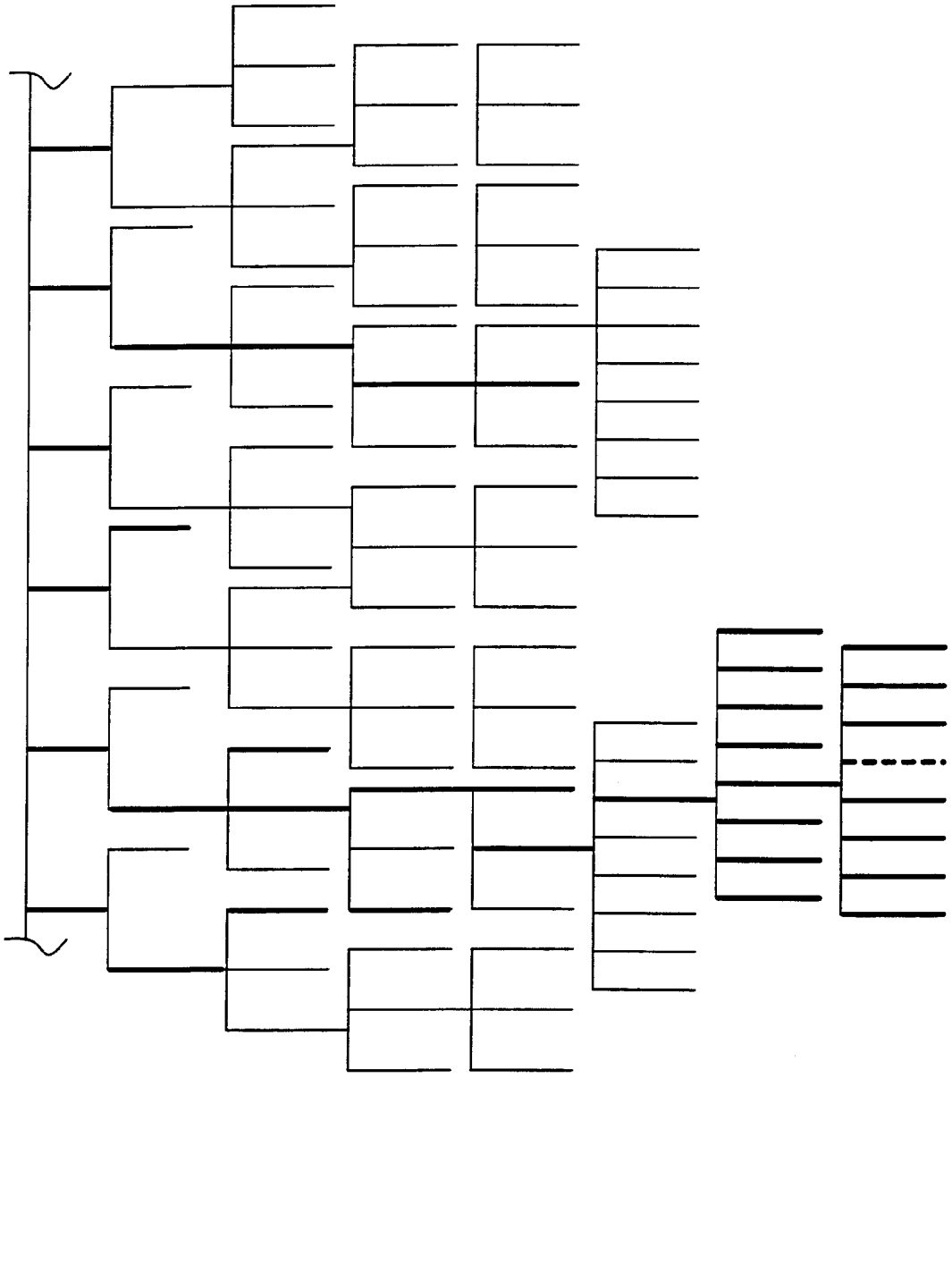


FIG. 7